SC71013.FR

(2)

DISTRIBUTED PROCESSING IN MULTI-HOP PACKET RADIO NETWORKS

Final Report for the Period
August 01, 1989 through June 30, 1991

Dr. A.R.K. Sastry
Principal Investigator

August 1991

Prepared for

U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211
Attn: Dr. James W. Gault, Electronics Division

ARO contract DAAL03-89-C-0017

Rockwell International Science Center
1049 Camino Dos Rios
Thousand Oaks, CA 91360

Approved for Public Release
Distribution Unlimited

Rockwell International
Science Center

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 09, 1991	Final Report, 08-01-89 to 06-30-91

4. TITLE AND SUBTITLE

Distributed Processing in Multi-hop Packet Radio Networks

5. FUNDING NUMBERS

ARO Contract
DAAL03-89-C-0017

6. AUTHOR(S)

A.R.K. Sastry (Principal Investigator), Robert Doyle and Iftikhar Shahnawaz

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Rockwell International Science Center
1049 Camino Dos Rios
Thousand Oaks, CA 91360

8. PERFORMING ORGANIZATION REPORT NUMBER

SC71013.FR

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U. S. Army Research Office
P. O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

ARO 25366.4-EL

11. SUPPLEMENTARY NOTES

The view, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report is concerned with the integrated evaluation of distributed processing scenarios in a highly dynamic environment such as a multi-hop packet radio network. To aid in this process, a simulation model was developed that has a network of identical packet radio nodes using spread spectrum random access protocols, an error model for multiple access interference and on-off partial band jammer, and end-to-end transport functions. An abstract description of a typical distributed processing scenario with hierarchical primary and subtask structures has been devised based on a 'script' that specifies processing time, sequences of subtasks (distributed over other nodes) and lengths of request and response messages and was integrated with the packet radio simulation. Extensive numerical results, primarily in the form of percent of successfully completed tasks and the mean task delay, were obtained through simulation runs representing a wide variety of task scenarios. Different processing scenarios such as assignment of task deadlines, levels of task-subtask hierarchy, and redundant processing of a subtask on multiple hosts, have also been examined.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Distributed processing, Multi-hop packet radio network, Spectrum random access protocols, Partial band jammer, Simulation, Primary and subtask structures, Task deadlines, Redundant processing.	66
	16. PRICE CODE

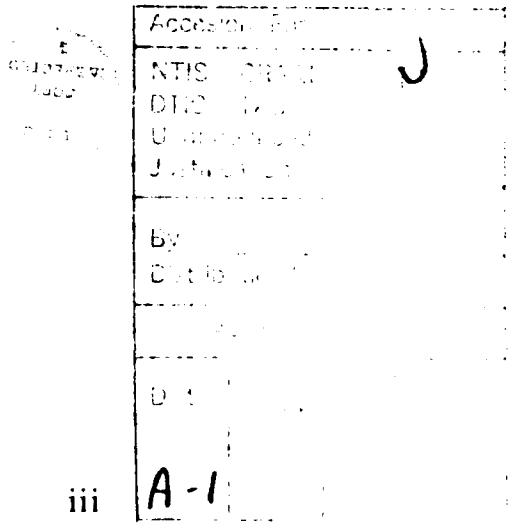
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

DISTRIBUTED PROCESSING IN MULTI-HOP PACKET RADIO NETWORKS

Final Report: ARO contract DAAL03-89-C-0017

CONTENTS

LIST OF APPENDIXES AND FIGURES



DISTRIBUTED PROCESSING IN MULTI-HOP PACKET RADIO NETWORKS

Statement of the Problem and Summary of Results:

Distributed processing is assuming an increasingly important role in defense applications. Multi-hop packet radio networks are expected to play a key role in supporting future distributed applications for mission control, logistics, and weapon system coordination. A proper assessment of the impact of the underlying communications on the performance of distributed processing missions and tasks is extremely important to understand the design issues, particularly for systems which must handle real-time or time-critical applications. It is important to develop an integrated approach to represent the interaction of application processes and the underlying communications protocols and networks.

This report is concerned with the evaluation of distributed processing scenarios in a highly dynamic environment such as a multi-hop packet radio network. To aid in this process, a simulation model has been developed that has a set of (user selectable number of) identical packet radio nodes, employs spread spectrum random access protocols, and has an error model that includes the effects of interference from concurrent transmissions as well as that induced by an on-off partial band jammer. To assure reliable communication, transport functions such as end-to-end acknowledgments and a limited number of retransmissions per packet are also included. An abstract description of a typical distributed processing scenario with hierarchical primary and subtask structures has been devised wherein a distributed task is described by a 'script' that specifies processing time, sequences of subtasks (distributed over other nodes) and lengths of request and response messages. In this description, a distributed task consists of a primary task and a hierarchy of one or more subtasks. The distributed processing representation methodology has been integrated with the packet radio simulation.

We obtained extensive numerical results through simulation runs representing a wide variety of task scenarios. Results are primarily in the form of percent of tasks that successsfully completed and the mean task delay in various cases with and without the presence of jamming and background packet traffic. A number of other statistics at the transport and link layers have also been collected. Different processing scenarios such as assignment of task deadlines, levels of task-subtask hierarchy, and redundant processing of a subtask on multiple hosts have also been examined.

1. INTRODUCTION

Multi-hop packet radio networks are expected to play a key role in providing communications service to the future Army tactical mobile missions in AirLand battle scenarios. Such networks are also very likely to be required to support distributed applications for mission control, logistics, and weapon system coordination. A proper assessment of the impact of the underlying communications on the performance of distributed processing missions and tasks is extremely important, particularly for systems which must handle real-time or time-critical applications. This report describes results of investigations in this area under the ARO contract DAAL03-89-C-0017 with emphasis on an integrated approach in representing the interaction of application processes and the underlying communications protocols and networks.

Distributed processing is assuming an increasingly important role in defense applications. Modeling and design of a distributed computing system is complex due to the need to represent the interactions among application processes and the associated communications. Current understanding of distributed processing and the associated communications is very limited due to lack of sufficient attention in the literature on (a) representation of traffic flows created due to the interactions among various processors and tasks, (b) integrated performance evaluation of protocols at various layers (for example, of the Open System Interconnection model (OSI) [1]), (c) procedures to handle special requirements such as time-critical traffic (to obtain desired response times).

Accurate representation of traffic flow at different functional layers and procedures to handle time-critical traffic are essential for any realistic evaluation of distributed processing systems. The evaluation tools should have reasonable level of detail so that results can be obtained for specific systems and yet be flexible such that they can be adapted for a wide variety of distributed systems. Several standardization efforts are under way to define an appropriate framework for describing models for communications in a distributed processing scenario [2],[3].

In a distributed process, an operation is performed by two or more cooperating entities. If, in addition, the entities are separated geographically, the process is said to be disbursed. In the present context we assume that the process is both distributed and disbursed. The effectiveness of such a process depends strongly on communications. The goal of the effort described herein is to investigate the efficacy of

providing this communications support with a packet radio network in view of the attendant unreliability of these systems.

Thus, we consider an application that is distributed and disbursed over a number of entities connected by a packet radio network. One such entity may initiate tasks at a number of remote locations by sending appropriate messages. Depending on the nature of the task, the message may contain instructions to be carried out, data to be processed, or a request for data. In any event, the task will require some processing and a response of some sort. The processing may require initiating subtasks at other remote locations and waiting for the corresponding response. The importance of communications to the success of this process is clear. To investigate this problem, we have developed an abstract description of a distributed process and coupled it with a simulation of a packet radio network, as an example.

In this work, we defined a multi-hop packet radio network with spread spectrum random access protocols and a methodology to describe distributed task scenarios and developed an integrated simulation tool to evaluate their interactions. Due to the complex nature of the problem and the difficulty in representing it through satisfactory analytical models, we leaned heavily towards simulation.

In the modeling of multi-hop packet radio networks, we considered use of spread spectrum (both frequency-hop and direct sequence types) random access (such as ALOHA) protocols (also commonly known as Code Division Multiple Access with ALOHA, i.e., CDMA-ALOHA). At the physical and link layers, we incorporateed in our simulation model the equivalent error rates expected as a function of parameters such as jamming level, error correction, number of users, etc., based on our own previous work and that available in the literature. Routing and flow control are important aspects to be considered at the network layer. Due to the topological and traffic dynamics of tactical packet radio networks, it is quite important to consider end-to-end transport functions through appropriate timers, number of acknowledgments and message segmentation. Finally, to represent the higher application-oriented layers, we defined the distributed processing needs of a mission as a set of tasks and subtasks that are assigned to different packet radio nodes that are executed asynchronously.

In section 2, we will introduce some of the terminology of packet radio network protocols and the related work in the literature. In section 3, we describe the work carried out on development of a comprehensive

simulation model for multi-hop packet radio networks. Section 4 describes the methodology for modeling the task structure for distributed processing applications, deadline assignments, and redundant processing for survivability. Section 5 gives numerical results from a wide variety of simulation runs and processing scenarios to illustrate the interrelationships among various parameters. Section 6 gives concluding remarks and the planned future work.

2. MULTI-HOP PACKET RADIO NETWORK PROTOCOLS - BACKGROUND

Design of a multi-hop store-and-forward packet radio network with mobile stations is quite complex due to the changing connectivity environment [4]-[11]. Thus, a number of heuristic procedures for retransmission control, congestion control, updating of routing tables, etc. become necessary, as can be found in the existing literature on packet radio networks (mostly related to the packet radio program of the Defense Advanced Research Project Agency). Since such procedures are not easily amenable to analysis, simulations and hardware/software testbeds are required to obtain a meaningful assessment of performance of the networks. Such an evaluation is particularly important for distributed processing applications using packet radio, such as ADDCOMPE [9].

A packet radio network consists of a number of nodes (packet radios, or PRs), each having a limited transmission range. All nodes use a common frequency and same bandwidth and contend for access to the channel, which can result in collisions. In general, corresponding transmit-receive pairs may not be able to communicate directly due to the limited range of each transmitter, requiring store-and-forward operations through intermediate nodes. Every node in the network is capable of acting as an intermediate node and keeps a routing table based on available `good-neighbor' nodes. The routing tables are updated using control information transmitted by each node periodically.

2.1 ALOHA and Related Schemes: In the simplest of the random access schemes, generally now known in the literature as the ALOHA scheme, a station transmits as and when it has a packet ready, which may collide in time with packets of other stations. Each station, if its packet suffers a colllision, reschedules it for transmission after a random delay, drawn from a distribution common to all the stations. The start times of the collective traffic from all such stations, consisting of the original and retransmitted packets, can be reasonably modeled by a Poisson distribution. Theoretical results show [12],[13] that a single-hop system with many users has a maximum channel utilization of about 0.184 when the packets have constant lengths. Since the starting times of the packets can be different, the collision intervals typically exceed the packet length and can even be of the length of several packets if more of them collide in a chain.

2.1.1 Slotted ALOHA Schemes: In a slotted ALOHA scheme, the start times of packet transmissions of all stations should coincide with the beginning of a time slot that is established through synchronization from a common clock. In this case, when two or more packets collide, they completely

overlap in time. Theoretical analyses show that [12],[13] this scheme has a maximum channel utilization of 0.368 for a single-hop network. The requirement of slot synchronization introduces the first element of complexity in the system compared to pure ALOHA, particularly if the nodes are mobile and thus have varying propagation delays among them. A centralized station serving as a reference synchronization control station may not be acceptable from the point of view of reliability. Even from the point of view of routing and flow control, a centralized approach may not be desirable. In a ``stationless" approach, the slot timing reference is to be derived in a distributed manner meeting the degree of accuracy required. It should also be noted that the above results on maximum channel utilization are derived for an "infinite" number of users.

For systems with a finite number of users, a Binomial distribution is a more appropriate model for the number of packets that arrive in a slot. As the number of users becomes fairly large, the Poisson assumption becomes more valid. Also, all the users are considered ``identical" with regard to their packet generating characteristics. However, if the users do not have identical traffic characteristics, the maximum utilization would be somewhat higher than that for identical users, a phenomenon termed "excess capacity" [13] with regard to the allowable message rate partitions among the users. Thus, the degree of nonhomogeneity in traffic rates should be considered in a network with a limited number of nodes and dissimilar traffic.

The communications capacity is maximized by requiring that all message transmissions are of equal length, but variable length transmissions are possible at lower capacity. Requiring all messages to have the same length is unrealistic, as message length will be a function of the information contained. One way around the fixed length constraint is to use ALOHA for a reservation subchannel, and then use a separate data channel on a reserved basis for the variable length messages. However, this approach is more susceptible to errors, particularly in a distributed control architecture, and has the drawback of an additional reservation delay.

2.1.2 Carrier Sense Multiple Access Schemes (CSMA): In this scheme [12], each station ``senses" or monitors the channel for any ongoing transmissions. Collisions can still occur if two or more stations sense the channel idle within a span of their relative propagation delay and transmit their packets. In a slotted system, sensing is done in a `minislot' corresponding to the largest propagation delay. Several variations are possible depending on how the stations respond to the state of the channel. In the simplest case, known as the non-persistent CSMA, if a station finds

the channel idle, it transmits a packet. If the channel is busy, it reschedules its transmission using a random delay drawn from a common delay distribution. In p-persistent CSMA, when a station finds the channel busy, it waits till the channel becomes idle. When it finds the channel idle, it transmits the packet with a probability p or delays it by a minislot with a probability $1-p$. In both the cases, if a station's transmitted packet suffers collision, it reschedules the packet for retransmission after a random delay (and again goes through the cycle of sensing etc.). In CSMA with collision detection (CSMA/CD) [14], if a station encounters collision during the transmission of a packet it suspends the transmission of the remaining portion of the packet and sends an erasure message. The throughput-delay performance of CSMA or CSMA/CD is very much sensitive to the ratio of the maximum propagation delay to the packet transmission time [12]. Equipment delays could create a situation equivalent to the presence of very large propagation delays and the CSMA operation may become very inefficient or may not even be feasible.

2.1.3 Stability Considerations: In addition to throughput and delay, stability is another dimension to the performance characterization of the contention based multiple access schemes. The statistical fluctuations in the traffic levels can cause a scheme to gradually drift into an unstable region of operation causing breakdown. Theoretical and simulation results for slotted ALOHA and CSMA show [15]-[18] that for an infinite population the stationary stable operation does not exist over an extended time, while for a finite number of stations the performance can degrade to unusable levels in a time (called first exit time [15]) that depends on the mean retransmission delay. Dynamic control procedures have been derived using Markovian decision models for slotted ALOHA with a finite number of stations [16],[17] to keep the system in a stable region. Only recently [19] have retransmission control schemes been devised that achieve stable ALOHA performance without constraints on the size of the user population (although, of course, the aggregate network traffic load is constrained). Accounting for the approximately Service-In-Random-Order (SIRO) nature of ALOHA, we have developed dynamic control procedures and determined the exact delay performance under such a scheme [20]-[22]. Based on these results, we developed dynamic control schemes for spread spectrum random access schemes [23]-[25].

2.2 Spread Spectrum Schemes

Spread spectrum techniques [26], [27] are used in some form or other in military communications for anti-jam and low probability of interception (LPI) purposes. Their inherent capability for multiple access will be an

additional attractive feature, particularly in the context of packet radio networks. With the increasing need to communicate with mobile stations and handle data traffic that is bursty (i.e., with rates that have a high peak-to-average ratio), packet-switched random access communications [12] on broadcast radio channels are becoming increasingly important to defense communications. Combination of random access schemes with spread-spectrum techniques is thus very attractive, since packets that collide in time can be retrieved to some extent [28]-[34]. Use of spread spectrum in random access schemes also allows superposition of acknowledgment traffic on the same channel with only a marginal degradation in the overall throughput [35].

Spread-spectrum-random access schemes can be implemented in a number of ways; slotted or unslotted, direct sequence (DS) or frequency hopping (FH), fixed or variable packet sizes, static or dynamic DS/FH assignment, with or without co-channel acknowledgment traffic, etc. When each receiver is assigned a unique pseudo random (PN) sequence to realize a DS or FH based multiple access operation, the resulting scheme is known as a Code Division Multiple Access (CDMA) scheme. The code set is to be chosen to meet low cross-correlation and high auto-correlation requirements for low error rates [36]-[38]. In packet radio applications, synchronization of the PN code should be achieved at a receiver within a reasonable fraction of the packet length [26],[39] and with low false synchronization probability. By changing the assigned spreading code (or code set) after a time slot in a prearranged manner, it is possible to realize a code-slotted operation [4] for anti-jam purposes. The objective is to devise a scheme that meets the throughput, delay, reliability, and anti-jam requirements, while keeping the bandwidth low and system implementation simple. Obviously, a number of trade-offs are involved in configuring a particular scheme for a given application.

2.3 CDMA-ALOHA Schemes for Packet Radio Networks

CSMA schemes are known to offer considerable improvement in throughput over the ALOHA schemes for small propagation delay [12] and thus are considered strong candidates for packet radio applications [28],[40],[41]. However, the requirement of carrier-sensing can lead to more jamming vulnerability in some applications, since the jammer also can sense the channel to effectively time his jamming signal so that messages can be interfered with at a high probability. This problem can be particularly severe if the sensing is done by the user at RF carrier level without the benefit of the security obtained through sensing after despreading a CDMA signal. Further, equipment delays due to modem turn

around times between transmit and receive modes, and processing times for implementation of higher level protocols can significantly reduce the effectiveness of CSMA in some practical situations. If these delays are large, their combined impact is such that the channel appears effectively as a high delay channel. When the delay corresponds to about a quarter of a packet length or more, the CSMA performance worsens compared to that of ALOHA [12]. Thus, CDMA-ALOHA schemes appear to be very attractive for packet radio networks with mobile stations. The difference in performance between slotted and unslotted versions of CDMA-ALOHA is so small that slot synchronization problems can be avoided by using unslotted CDMA-ALOHA [42].

2.4 Routing

There is a great deal of difference in the characterization and modeling of single-hop and multi-hop packet radio networks due to the need for routing and store-and-forward functions in the latter. Tobagi [43] analyzed a centralized two-hop system for ALOHA and CSMA schemes. The problem formulation in a fully distributed multi-hop mobile radio network is more complex [44]-[47]. When multiple hops are involved, the development of a traffic model becomes quite complicated. If the stations are mobile, it may be assumed that each station is a potential relay. A packet received at a given station may either belong to that station itself or may need to be forwarded to another destination. Thus, the packet traffic flowing through a station (node) cannot be characterized in isolation. Also, it may be advantageous to have two queues, (i) for the packets originated locally and (ii) for packets in transit. The latter queue may be given priority for transmission as the packets in the queue already used up some network capacity and further delay might impact more on traffic handling by other nodes in the network.

In a multi-hop packet radio network, many routing schemes are possible. In the DARPA packet radio protocols [6], at each node, the route of a packet and the route length are maintained in a routing table which is periodically updated using network control packets. Initially, each node broadcasts packets without having any knowledge about its neighboring nodes. When a node receives a control packet from its neighboring node, it adds the transmitting node of the packet to its list of aware-of-nodes and sets the route length from it to that node to 1 if the route length is unknown. Each node also periodically broadcasts a control packet that carries its running transmission count, current receive link rating for all its neighbors, its aware-of nodes, and the corresponding route lengths such that all nodes receiving the control packet can update their respective local information.

After transmission of a control packet, each node waits a certain time interval (same for each node) before broadcasting its next control packet.

Several alternative approaches to routing such as MFR (Most Forward with Fixed Radius) strategy [46] are also feasible in multi-hop networks. Appropriate performance measures need to be developed for effective comparison of different routing schemes.

2.5 Pacing

Pacing provides flow and congestion control by applying a function of the measured delay to separate successive transmissions to each neighboring node. The detail of this conversion of forwarding delay to pacing delay and the mechanism for measuring or estimating forwarding delay is given in Ref. [6]. When the updated forwarding delay grows beyond the maximum allowable delay, this maximum value is then used for the new delay. If this happens and if no acknowledgment is received with all the retransmissions reaching the allowed limit, the link connectivity rating is marked bad and all routes with the same next node are also marked bad. All the packets in the transmit queue that are going to that next node will then be broadcast only once when they are selected, with no further forwarding to that unreachable next node.

We have given a brief description of packet radio networks in this section to highlight the issues in the design of such networks. In the next section, we will describe in detail the specific features incorporated in a simulation model that we developed for a multi-hop packet radio network. The simulation model is used in our investigations to understand the interactions between the highly dynamic traffic features of the network and its ability to support distributed processing needs of different application scenarios.

3. SIMULATION OF A MULTI-HOP PACKET RADIO NETWORK

In our simulation model for a packet rdaio network, we considered spread spectrum random access schemes, specifically CDMA-ALOHA protocols. CDMA schemes typically employ receiver-directed codes, where each node's receiver is assigned a unique code sequence that is orthogonal to other codes in the set, so that multiple access interference can be substantially reduced. However, there is a short window at the start of the reception of a packet at a receiver, during which a collision is possible if multiple transmissions are simultaneously directed to this node. If the window duration is a small fraction of the packet length, collisions can be greatly reduced. However, if the receiver has already captured a packet, a second packet directed to that receiver, while usually not interfering with the reception in progress, will not be received and will require a subsequent retransmission. Thus, though significantly reduced, part of the common channel interference remains. In addition, effects of jamming need to be considered.

At the physical and link layers, we incorporate in our simulation model the equivalent error rates expected as a function of parameters such as jamming level, error correction, and number of users. Routing and flow control are important aspects to be considered at the network layer. Due to the topological and traffic dynamics of tactical packet radio networks, it may be quite important to consider end-to-end transport functions through appropriate timers, acknowledgments (ACKs), and message segmentation. In this section, we will describe the packet radio network simulation. The simulation is written in C and is hosted on a SUN workstation. Representation of distributed processing (higher application-oriented layers) is described in the next section in terms of a set of tasks and subtasks that are assigned to different packet radio nodes and executed asynchronously.

3.1 Network Simulation

The network portion of the simulation consists of models of the packet radio nodes and node interface modules, background noise and jamming, and routing methodology. Figure 1 gives the schematic of data flow.

3.1.1 Packet Radio Node

The node model consists of four modules: (1) a processor module which determines the disposition of incoming packets, (2) a router module which determines proper routing for all packets to be transmitted, (3) a flow

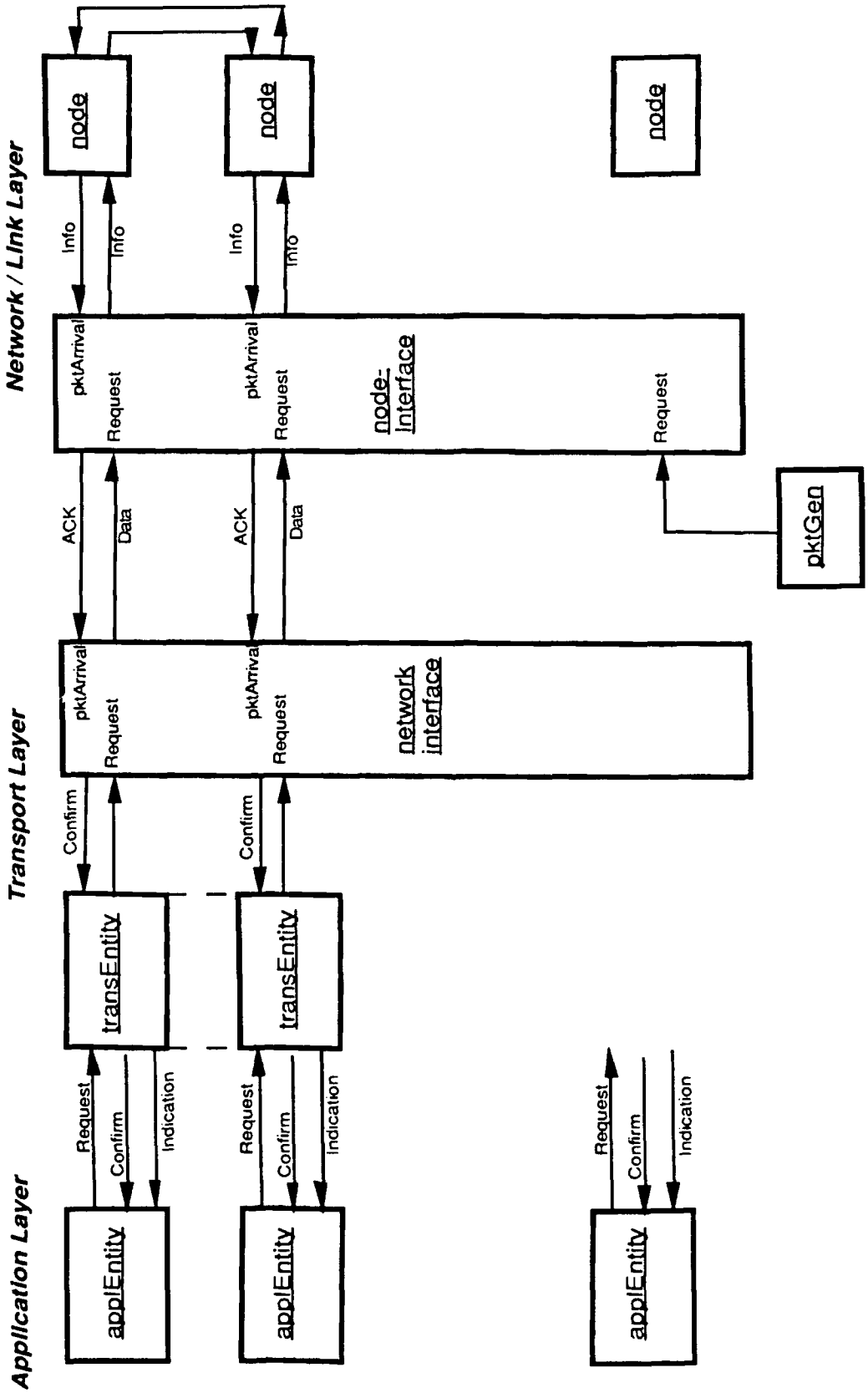


Figure 1. Data flow model for distributed processing /packet radio network

control module which controls timing between successive transmissions, and (4) a transceiver module which models packet transmission and reception. Figures 2-4 give the packet radio block diagram and the state machines for the transmitter and receiver.

Processor Module. The processor module determines the disposition of all locally generated and incoming packets with the following functions:

(i) Accept remote packets from the transceiver module, and locally generated packets from the node interface module,
(ii) Send packets to the router module for update of aware-of and neighbor node lists, and
(iii) Determine disposition of received packets as follows:

Not addressed to this node - discard packet.

Control packet - send to the router module for routing analysis and update.

Positive ACKs - remove ACKed packet from the transmit queue.

Local destination - acknowledge (create ACK packet and send to router module) and send packet to the node Interface module.

Remote destination - ACK the packet and pass it on to the router module for onward transmission.

Router Module. A router module determines proper routing (designation of 'next' node) for all packets to be transmitted, updates its routing database in accordance with received control packets, and generates control packets to be broadcast to neighboring nodes. It performs these functions:

(i) Accept packets from the processor module,
(ii) Update the routing database in accordance with received control packets,
(iii) Periodically create and pass to the flow control module a control packet to be flooded to all neighbors,
(iv) Provide routing information (next node) for forwarded and locally generated packets, and
(v) Pass packets to the flow control module for transmission.

Flow Control Module. A flow control module maintains a transmit queue in accordance with the established priorities, determines the order in which

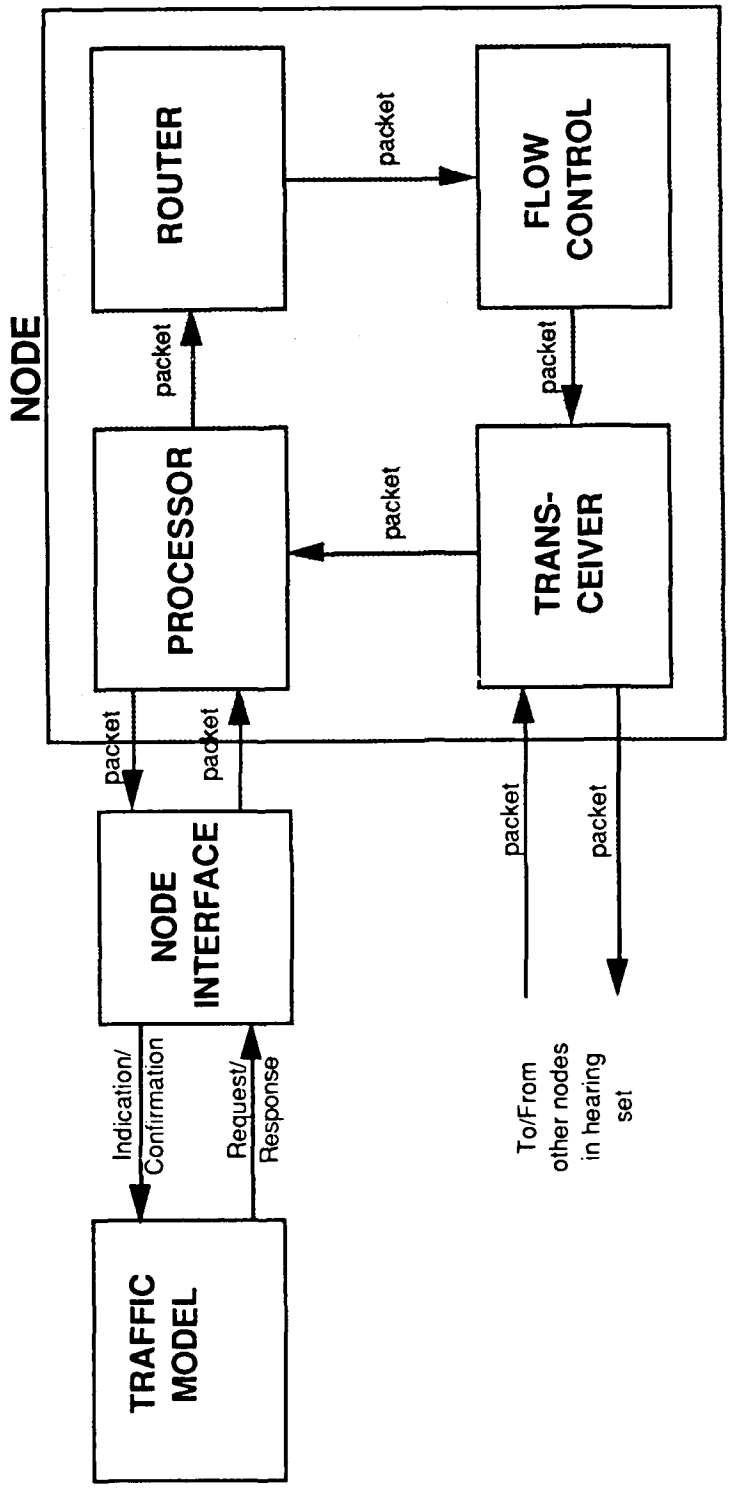


Figure 2. Packet radio simulation - Block diagram

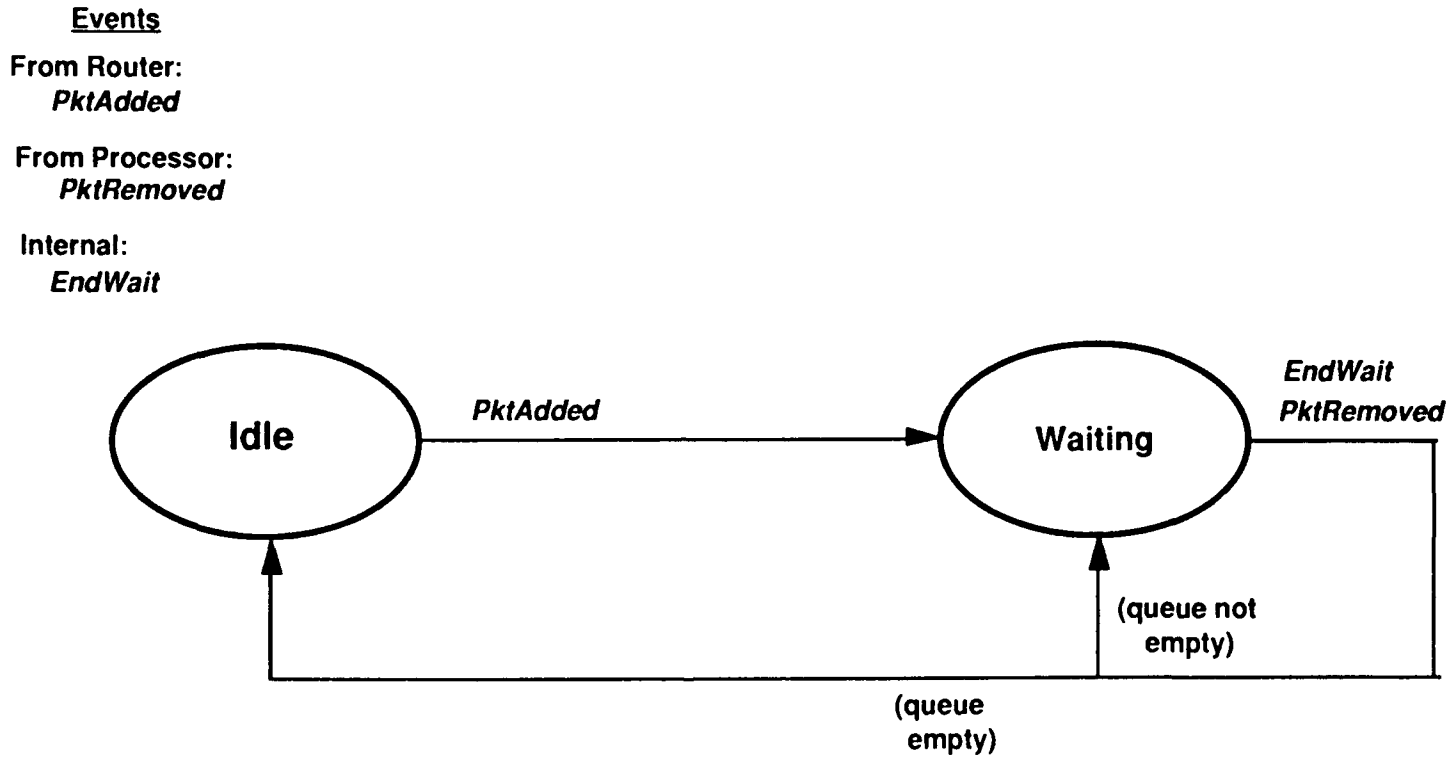


Figure 3. Flow control state machine

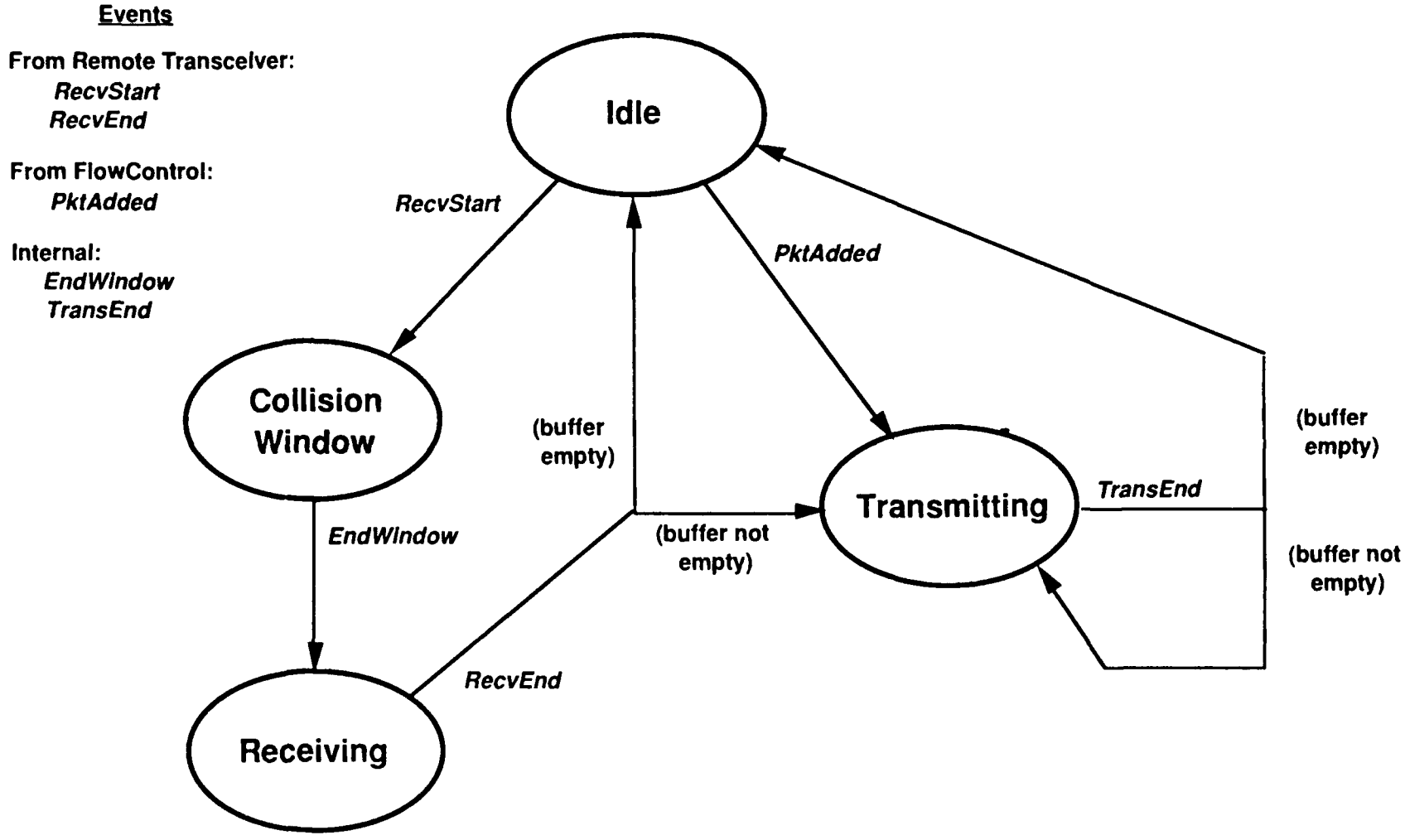


Figure 4. Transceiver state machine

packets are to be transmitted, and controls timing between successive transmissions. The flow control module manages the node packet transmission rate through the following functions:

(i) Accept packets to be transmitted from the router module,
(ii) Maintain the transmit queue in accordance with established priorities as to packet type, 'next' node and any other,
(iii) Determine the order in which packets are to be transmitted,
(iv) Control timing between successive transmissions, and
(v) Pass packets to the transceiver module for transmission.

Transceiver Module. A transceiver module, which simulates the spread-spectrum receiver-directed protocol, transmits packets to neighboring nodes (with propagation delay), receives neighboring node transmissions, and determines validity of received packets (collided or not) and jamming effects. The transceiver module provides the interface with the rest of the network. It performs the following functions:

(i) Accept packets from the flow control module for transmission,
(ii) Transmit information and ACK packets to indicated 'next' node after appropriate propagation delay,
(iii) Transmit control packets to all neighbors,
(iv) Receive packets from neighboring node transmissions,
(v) Determine clashes between received packets,
(vi) Determine effects of jamming and other noise on received packets, and
(vii) Forward valid packets to the processor module.

Node Interface. The node interface module permits the packet radio simulation to support a variety of traffic generation models with varying levels of detail. It translates protocol interaction primitives to and from the packet radio packet format. The simulation provides two sources of traffic. A background traffic generator, discussed below, provides random traffic. The distributed application, discussed in the next section, defines the traffic related to the performance of a specific sequence of tasks. The node interface module performs the following functions:

(i) Accept Request and Response primitives from either traffic source,
(ii) Create appropriate packet and send to the processor module,
(iii) Accept arrived packets from the processor module, and
(iv) Send Indication or Confirmation, as appropriate, to the traffic source.

3.1.2 Routing

In the development of this model, use of a minimum hop routing policy is assumed. Initial routing is established by the following algorithm. From transmission range and node location, the neighbors of each node are determined, and thus all one-hop routes are determined. Then all two-hop routes are found from neighbor of a neighbor consideration. In general, if there is a n-hop route from x to y, and z is a neighbor of y, then there is a n+1-hop route from x to z, with the same 'next' node as the route from x to y. Having found all n-hop routes, all n+1-hop routes are found using this relationship. Thus, three-hop, four-hop, etc., routes are found in sequence.

For static routing, the router module needs only a table of 'next' nodes, one for each possible network destination. However, if the routing need be adjusted dynamically, the router also needs to know the length, in hops, of each route, so it can judiciously select a new route if the old one becomes no longer valid.

Routing tables are established initially as described above and are updated periodically from information in control packets received from neighbors.

3.1.3 Background Traffic

The background traffic generator provides random traffic to the packet radio network. Such traffic is in addition to that generated by the distributed application. The background generator provides a means to vary the load on the network in a controlled manner.

A simplified generator is used. An input parameter specifies the desired packets per second per node. From this the average packet intergeneration time is determined. The next packet is generated after a random delay is drawn from an exponential distribution. Two additional uniformly distributed random numbers are drawn to select packet source and destination, and a request primitive is sent to the node interface module which creates the packet and initiates transmission.

3.1.4 Jammer Model

The jammer model incorporated into the packet radio simulation includes the effects of interference from concurrent transmissions as well as those induced by an on-off partial band tone jammer. Through external event scheduling, the jammer may be dynamically assigned to a specific node or

set of nodes. Following an initial short 'capture' period, during which an interfering transmission will cause loss of a packet with probability one, the error model provides probability of packet loss taking into account jammer presence, interfering transmission in the same frequency bin, as well as the assumption of Reed-Solomon error encoding [38],[42]. Appendix I gives the formulae used to compute the error probabilities that were incorporated in the simulation model.

3.2 Transport Functions

As the distributed application depends on reliable end-to-end communication, a simplified model of the transport layer is included in the simulation. This is especially important because of the unreliable nature of the link level due to mobility and multiple user interference. Transport functions included are packetization of long messages, end-to-end acknowledgments and retransmissions when necessary. Our previous work on transport protocol modeling and analysis [48] was useful in this regard.

When an application element needs to send a message, it notifies the corresponding transport layer element providing the destination and message length. The transport layer divides the message into packets which it attempts to send sequentially. For each packet, a Request primitive is sent to the node interface module and a timer is set. The node interface module initiates transmission. If, and when, the packet arrives at the destination node, it is forwarded to the node interface module which sends an Indication primitive to the peer transport entity. The latter, in turn, sends a Response primitive to the node interface module which initiates transmission of a packet back to the original source. This packet constitutes a transport layer acknowledgment, but at the network level it is treated as any other information packet.

If, and when, this packet arrives at the source node, it is forwarded to the node interface module which sends a Confirmation primitive to the originating transport entity. If the Confirmation is not received before the timer expires, the packet is sent again. The time-out time and the number of retries are input parameters.

If, and when, all packets in the message have been received by the peer transport element, the destination application entity is notified. If, and when, all packets have been confirmed at the originating transport entity, the source application entity is notified. If, on the other hand, any packet fails after exhausting all retries, the source application entity is notified immediately and the message is aborted.

4. MODELING DISTRIBUTED PROCESSING SCENARIOS

This section describes the methodology adopted for modeling distributed processing scenarios, including deadline assignments for tasks and redundant processing for survivability. The resulting model has been integrated with the packet radio model discussed in the previous section. An abstract description of distributed processing has been devised in which a distributed task is described by a 'script' that specifies a request message length, processing time, sequences of subtasks (distributed over other nodes) and a response message length. In this description, a distributed task consists of a top-level task and a hierarchy of one or more subtasks as shown in Figure 5. A number of these task hierarchies may be active concurrently in the same network.

4.1 Task Hierarchy Model:

A subtask is hosted by an application entity at one or more packet radio nodes and is initiated by receiving a request message from some other host. It invokes zero or more subtasks sequentially or in parallel by sending request messages to the appropriate host or hosts. Thus a subtask's execution time is somewhat unpredictable due to subordinate subtask processing and communications delays. When all subtasks are complete, as indicated by receipt of appropriate response messages, and following a specified processing delay, the subtask sends its response message to the initiating node. The operation of a top-level task is identical except it is initiated directly so there is no request or response messages.

The request and response message lengths, the subtask sequences, and the processing time are specified by task script. Message lengths and time delays may be constant or drawn from various random distributions.

Figure 6 shows an example of a subtask time line. This subtask has two sequences of subordinate subtasks, processed in parallel. The upper sequence consists of two subtasks, processed sequentially; the lower sequence consists of a single subtask. Subtask processing is initiated at time T_0 by sending a request message which arrives at time T_1. Following (possibly random) processing delays, at times T_2 and T_3, respectively, the sequences are initiated by sending request messages. Subtask processing is delayed until T_6 when both the sequences have been completed. Following a processing delay, a response message is sent which arrives at the initiator at time T_8.

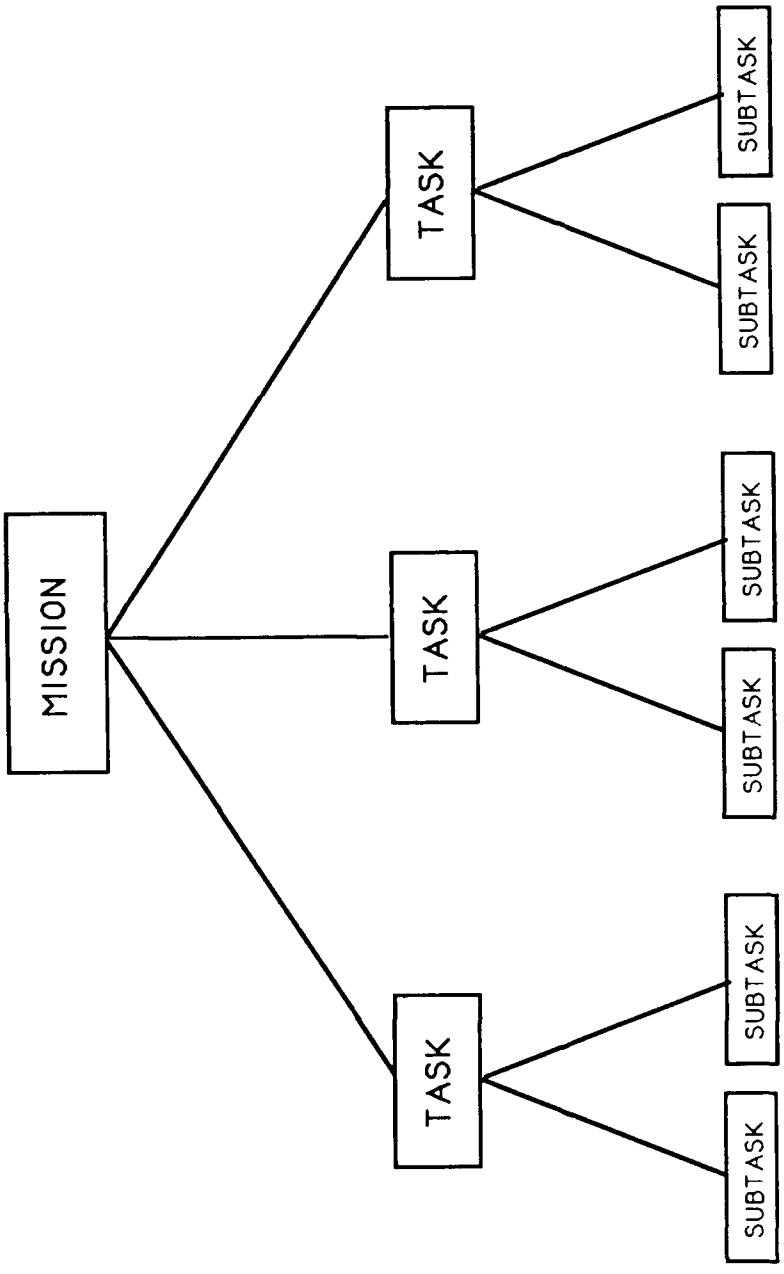
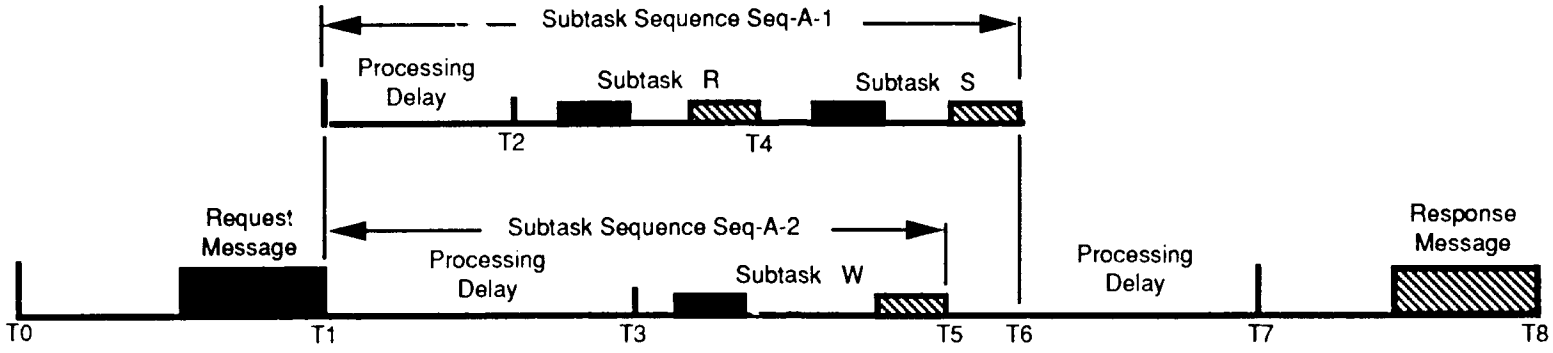


Figure 5. Tasks and Subtasks in a Mission



Task A Timeline

T0 Task A initiated from node x; request message sent to node 2.
T1 Request message for Task A arrives at node 2.
Subtask sequences Seq-A-1 and Seq-A-2 initiated.
T2 End of processing delay for Seq-A-1. Subtask R initiated.
T3 End of processing delay for Seq-A-2. Subtask W initiated.
T4 Subtask R complete; Subtask S initiated.
T5 Subtask W complete; subtask sequence Seq-A-2 complete.
T6 Subtask S complete; all subtask sequences complete.
Initate task processing.
T7 End of task processing delay; response message sent to node x.
T8 Response message received at node x; task complete.

Figure 6. Timeline Model of a Distributed Task with Subtasks

4.2 Task Queueing

It is assumed that a host can only perform one task at a time. If, while engaged in one task a host receives a request for a second task, the request is queued. When a host completes a task, it will initiate the next request in the queue, if any. The average time a task spends in a queue is provided in the simulation results.

Distributed task performance is impacted by contention for node resources as well as communication resources. In order to separate the communication problem from node contention, it is possible to design distributed task structures that use disjoint nodes. As a result, there is never any queueing delay and any task completion failures are due solely to communication problems. Analysis of such a task structure can give an insight in to the effects of the underlying communications network on distributed processing. A more realistic scenario, however, includes contention for both node and communication resources.

4.3. Assignment of Deadlines to Tasks and Subtasks

A task can recognize a communication failure in three ways: (1) it is notified by the transport layer that it cannot communicate a request to some subordinate subtask, (2) it receives a message from a subordinate subtask reporting a lower level failure, and (3) it fails to receive an expected response from a subordinate subtask in a specified time. The last case requires the imposition of a time-out on task completion. In fact, a time-out may introduce more operational realism: a task fails if it is not completed in some prescribed time. In any event, a time-out is the easiest and most straightforward method to detect communication failure. However, it may be desirable, in addition, for the subordinate subtask to report the failure to the subtask initiator.

Clearly, in a real case there exists some time limit on the completion of a task. From the simulation point of view, because of communication failure or loss of a node, some tasks may never finish. In order to gracefully recover from this kind of failure, and prevent the initiating host from waiting indefinitely on a failed subtask, it is necessary to impose a deadline on each task. There is the problem of determining a proper value for the time-out. If too short, the task will occasionally be aborted even when there is no failure; if too long, task completion may suffer an excessive delay.

In an actual scenario, the time limit is imposed at the top level. The top level task must then budget this allotted time between subordinate subtasks. These in turn, must budget their allotments to their subordinate subtasks. Thus deadlines are determined in a top-down fashion.

Figure 7 shows a simplified task timeline that will be used to illustrate how tasks deadlines have been implemented. The time to complete the top level task includes possible queueing delay (if the top level host were busy at the time the task was initiated), time to perform subordinate subtasks, and processing time. The figure indicates a single subtask. The time to perform this subtask includes the time to communicate the request to the subtask host, possible queueing delay, time to perform any required subordinate subtasks, processing time, and time to communicate the response to the initiating host. If the top level task must complete by the indicated deadline, the subtask must complete by a deadline that is earlier by its response communication time and the top level processing time. The top level task, when initiating a subtask, must provide this deadline for the subtask along with the request.

It is assumed that, at the time the subtask request message is sent, the processing time is known. The response communication time is unknown and must be estimated. This is done using the distance (hops) between top level and subtask hosts, the number of packets in the response, and an estimate of the communication delay per packet per hop. The latter is an input to the simulation. The subtask deadline is determined by taking the top level deadline and subtracting the processing time and the estimated communication delay. This value then is transmitted to the subtask host along with the request. Note that deadlines treated in this manner are absolute times, rather than delays.

If it is required to perform two or more sequential subtasks, the deadline allocation is more complex. A simple solution is to compute the deadline for the final subtask and impose it on each of the sequential subtasks. If two or more subtasks are to be performed in parallel, deadlines are computed for each and the latest (the most generous) is imposed on both.

In a similar manner the subtask would compute the deadlines for its subordinate subtasks.

When a deadline is imposed on a top level task or a subtask, a timeout event is scheduled at the deadline time. If the task completes and sends its response before this time, the timeout event is canceled. Otherwise, at the deadline time, the timeout event is processed and the task is aborted.

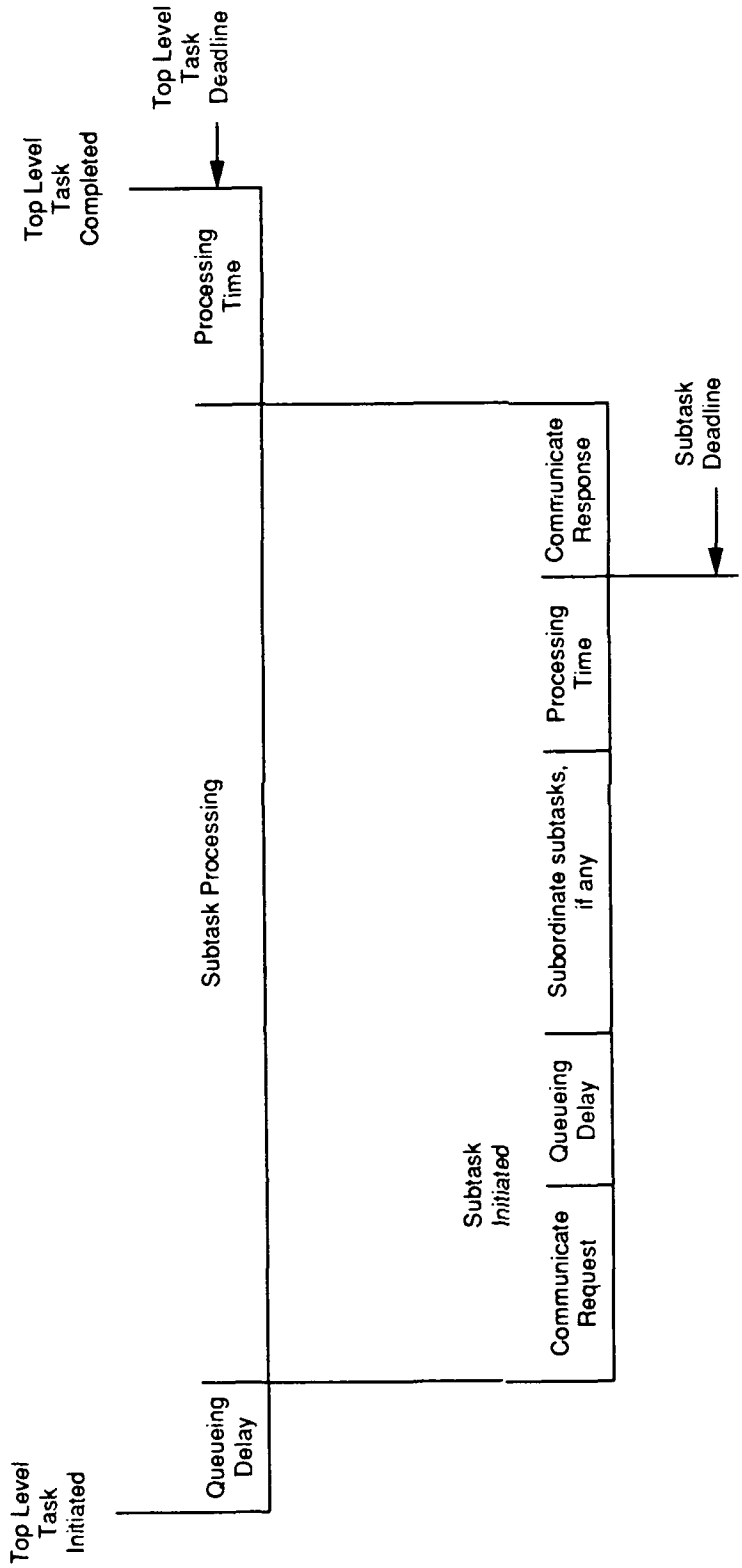


Figure 7. Task Deadlines.

The tasks under consideration in this study are completely abstract. As such, there is no specific basis from which deadlines can be established. In our study, deadlines are chosen merely for the purpose of illustrating the results. Task deadlines chosen must be reasonable: if too short, all tasks will be aborted; if too long, total task processing time becomes unreasonably long and the analysis becomes meaningless. In order to determine a reasonable set of task deadlines, it is noted that the uncertainty in time to complete a task is due to

communication delays.

(possible) statistical variation in processing times.

(possible) statistical variation in request and response message lengths.

It is necessary to make estimates of upper limits for these variables. Communication delays may be estimated by running the simulation, perhaps iteratively. Using such estimates and a knowledge of the task hierarchy structure, it is possible to estimate task completion times.

This procedure has been automated within the simulation program by writing a function that, given an estimate for communication delay, computes the estimate for task completion. This function invokes itself recursively, as required, to estimate subordinate subtask completion times. Using this function and an input estimated communication delay, the simulation can compute its own task deadlines. By running the simulation for several different values of communication delay and observing the number of tasks that complete after the deadline, a reasonable estimate of appropriate task deadlines can be obtained. In a real system, deadline for a top level task is derived from the related application or the mission.

4.4. Survivability Through Redundant Processing

Turning next to how the abstract application can respond to a communication failure, we can either accept partial task completion or postulate alternate methods of performing certain subtasks. In this case, two or more nodes will be specified to host some or all subtasks. Then when a subtask fails, it can be initiated on one of the alternate nodes. As this may impose excessive delays on task completion, we consider the use of redundant subtask processing. Here, the same subtask is initiated concurrently at two (or more) separate hosts. A response will be accepted from whichever host finishes first and the others will be ignored.

There are several variations on a scheme like this. For example, certain subtasks could be declared critical and initiated redundantly as described above. Other less critical subtasks could be retried at the same or other nodes once they have failed. Finally the failure of some subtasks could be ignored with respect to the completion of a superior task.

As described so far, the task structure depends upon each subtask in the hierarchy: if any subtask fails, the top level task will fail. Subtask failure can be due to communication failure or lapse of the allowed time. Task reliability can be improved by initiating tasks on multiple hosts.

Task redundancy has been implemented in the following manner. When more than one host is specified for a task, a request message is sent to each node in the list. The response is accepted from whichever host finishes first. Once one response has been received, responses arriving later are ignored.

Redundant operation can theoretically improve task reliability. However, the advantages of redundancy can be offset by the added communication congestion produced by the additional tasks.

5. NUMERICAL RESULTS

The distributed processing model, representing the equivalent of the application layer, has been integrated with the packet radio network model in the simulation program. A Performance Monitor module gathers statistical data on the operation of the simulation model. A number of simulation runs have been carried out under various scenarios to obtain numerical results described in this section. A typical output from a simulation run is shown in Appendix II.

5.1 Failed Tasks, TPDUs, and packets and their mean delays

Figures 8 and 9 show the results obtained for a 9-node network that compare performance as a function of transmission rate in terms of percent of tasks completed and average delay in processing a task (application level), percent of lost transport data units (TPDUs) and the average TPDU delay (transport level), percent of lost packets and the average packet delay (link level). Figure 8 shows the results as a function of packet radio transmission rate (16 and 32 Kb/s), for specified background packet generation rate, and jamming probability. At the lower rate of transmission (16 Kb/s for the example), the percent of aborted tasks, failed TPDUs, and lost packets are all higher than at the higher rate. The average delays are also higher for all the three cases compared to the corresponding values at 32 Kb/s rate. For this set of parameters, overall performance at 32 Kb/s rate seems to have improved. Figure 9 shows the results as a function probability of jamming in a symbol interval with the transmission rate at 32 Mb/s. The results indicate that consistently for all the three levels (application, transport, and link), degradation in the percent failures and average delay started after the jamming probability reached 0.05, indicating that the collisions and errors induced in the packets were correctable up to that level and the system began to breakdown. Since the number of parameters involved is quite large and their interrelationship is quite complex, careful examination over a number of scenarios will be needed to discern performance patterns.

All the rest of the results presented in this section were obtained using a 20-node packet radio network topology shown in Figure 10. The nodes are numbered 0 to 19. Distributed task descriptions will be presented in figures showing task hierarchical structure where each node of the tree represents a subtask. The diagram will indicate the network node or nodes hosting that subtask. Such node designation corresponds to the node identification number shown in Figure 10.

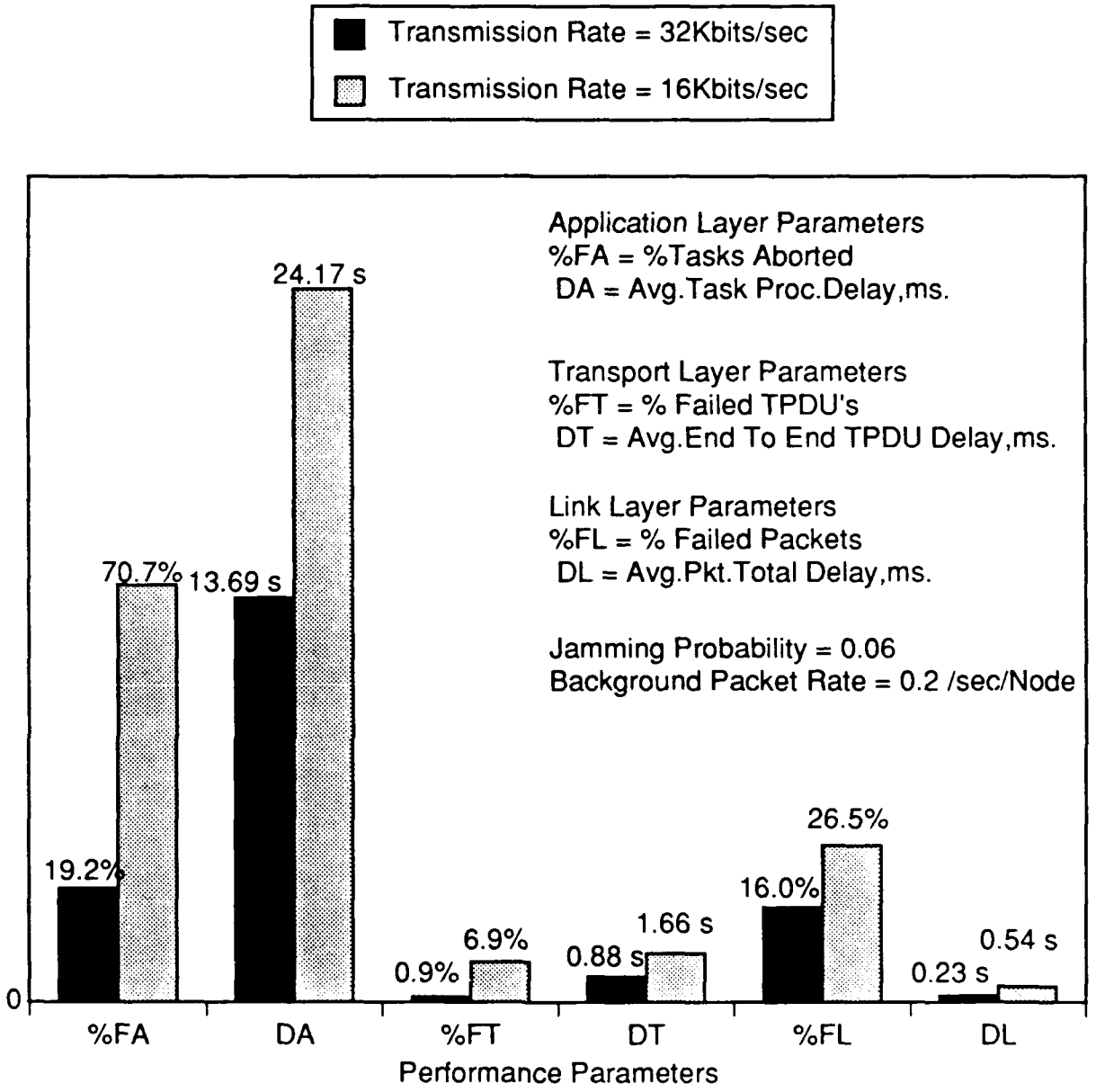


FIG. 8. PERFORMANCE PARAMETERS AS A FUNCTION OF TRANSMISSION RATE

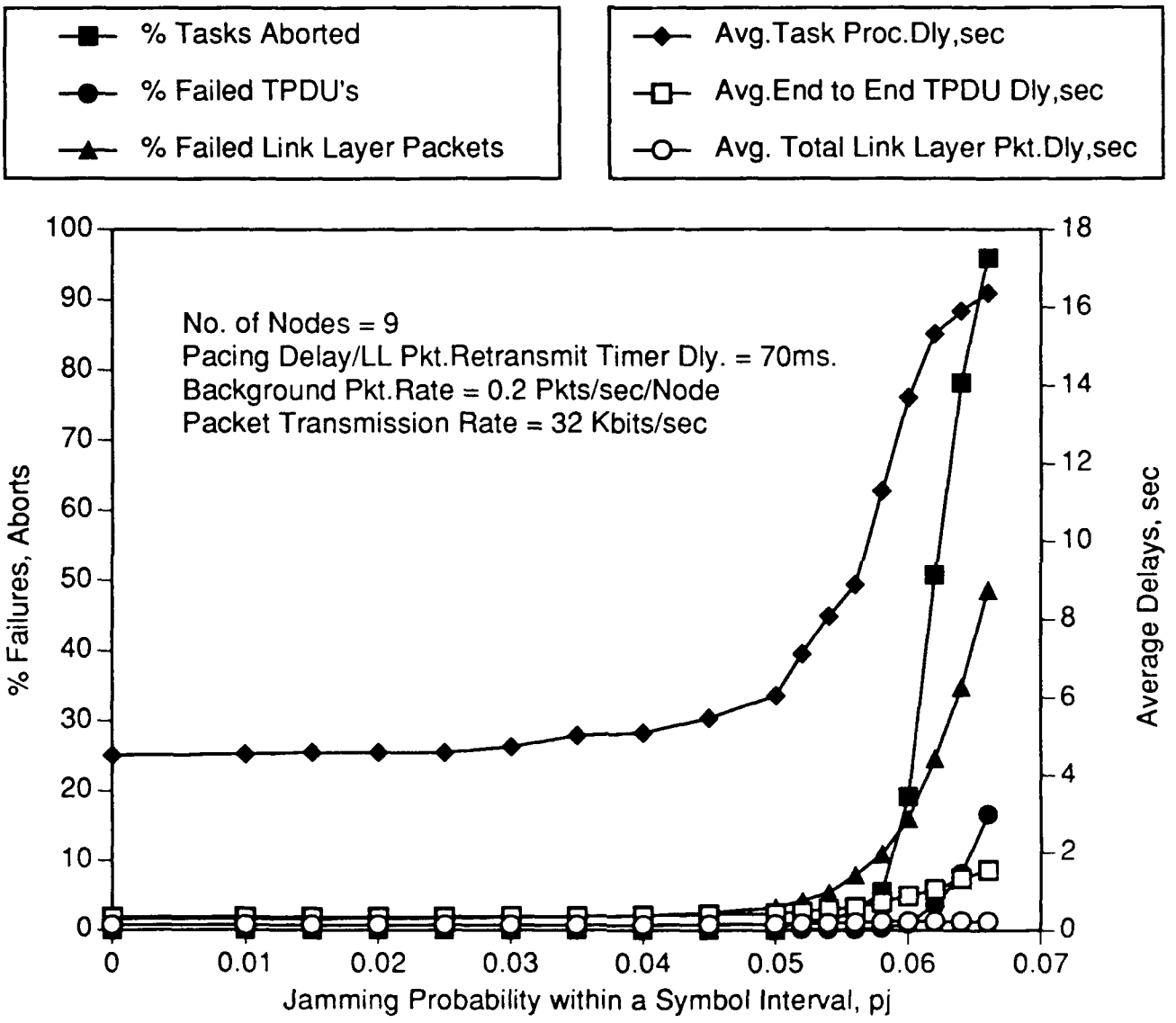


FIG. 9. PERFORMANCE AS A FUNCTION OF JAMMING LEVEL

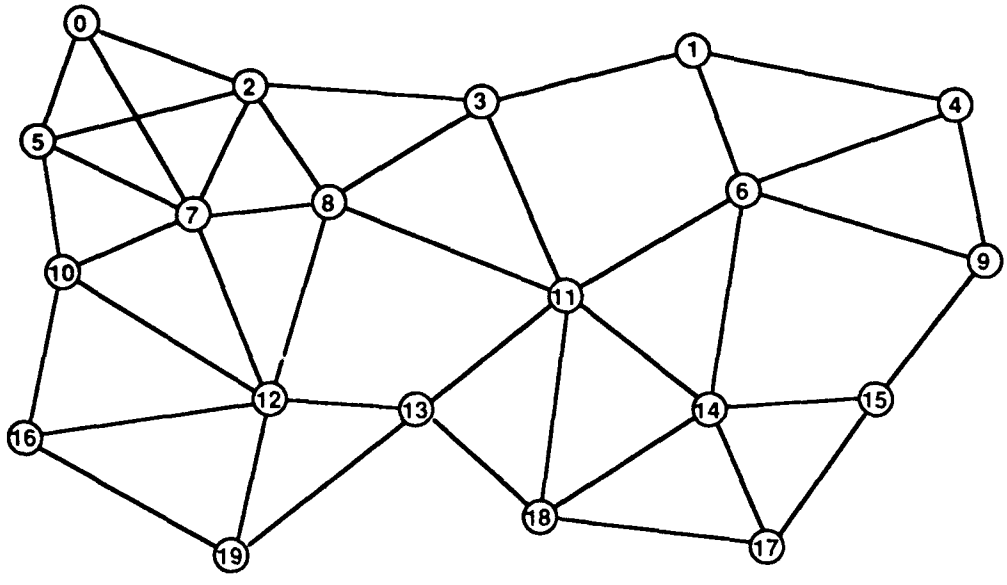


Figure 10. The network topology with 20 nodes, used in the simulation runs

5.2 Task Deadlines

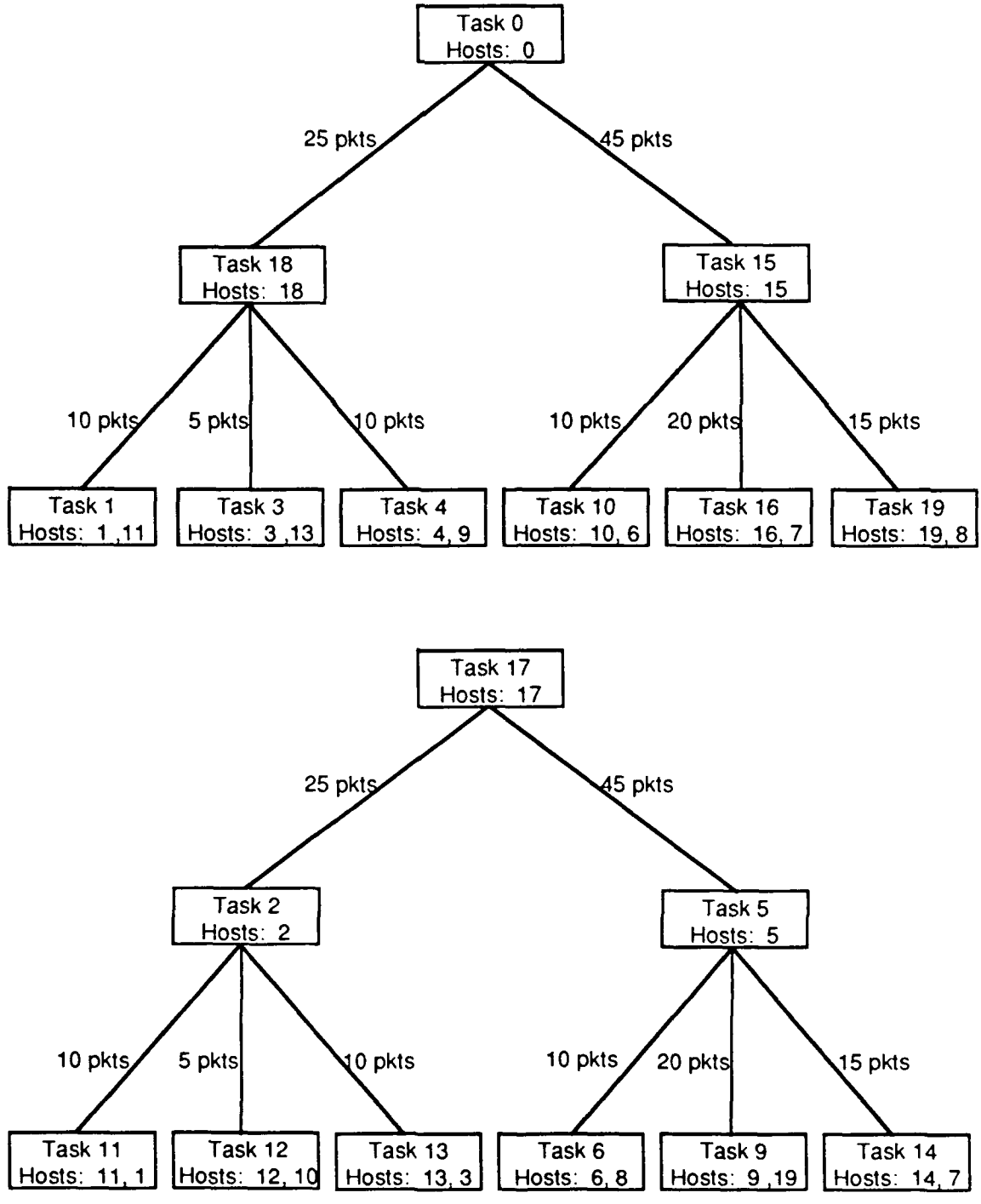
A number of simulation runs were made with task deadline as the independent variable. All of these runs were made using the task structure shown in Figure 11. This structure includes two similar three-layer hierarchies deployed on the 20 node network referred to above. The nodes hosting the various subtasks are identified and the required inter-task communication is indicated by number of packets. This number represents the subtask response. Though not indicated on the task diagrams, the subtask request was in all cases a single packet. In the third layer, two host nodes are specified for each subtask. Some of the runs to be discussed feature redundant operation of the third level subtasks. The indicated nodes were utilized in the order indicated: for no redundancy the first listed node was used, for single redundancy both nodes were used.

In the task deadline runs, the two task structures of Figure 11 were initiated asynchronously. The inter-generation time for each top level task was uniformly random between 150 and 250 seconds. This produces a random amount of interference between the two top level tasks. There was no jamming present for these runs.

In order to simulate various levels of network loading, background traffic has been introduced in some of these runs. This traffic is unrelated and is in addition to that incident to the active tasks, but is generated at random by a Poisson process. At each Poisson event source and destination nodes are selected at random, a packet is sent from the one to the other, and the next Poisson event is generated. The Poisson distribution parameter is selected to produce traffic corresponding to a user input rate in packets per second per node.

The task deadline runs cover a wide range of deadlines; from those that are so short that no tasks can complete to those that are so long that the additional time does not result in any further improvement. The first of these is shown in Figure 12, which presents percent top level tasks completed as a function of task deadline. These runs did not include redundant processing. The three curves represent background rates of zero, 0.1, and 0.215 packets per second per node.

It will be observed that a deadline threshold exists somewhere about 100 seconds; below this value it is impossible for the task to complete. As the deadline increases from this lower threshold, task completion rate rises sharply. After about 200 seconds no significant improvement is seen. The saturation value is seen to be an effect of network traffic levels. The



Processing Time = Uniform (5, 10) Seconds for each task.

Host node numbers are as indicated in the topology shown in Figure 10.

Figure 11. Task structures used for simulation runs as a function of deadline time

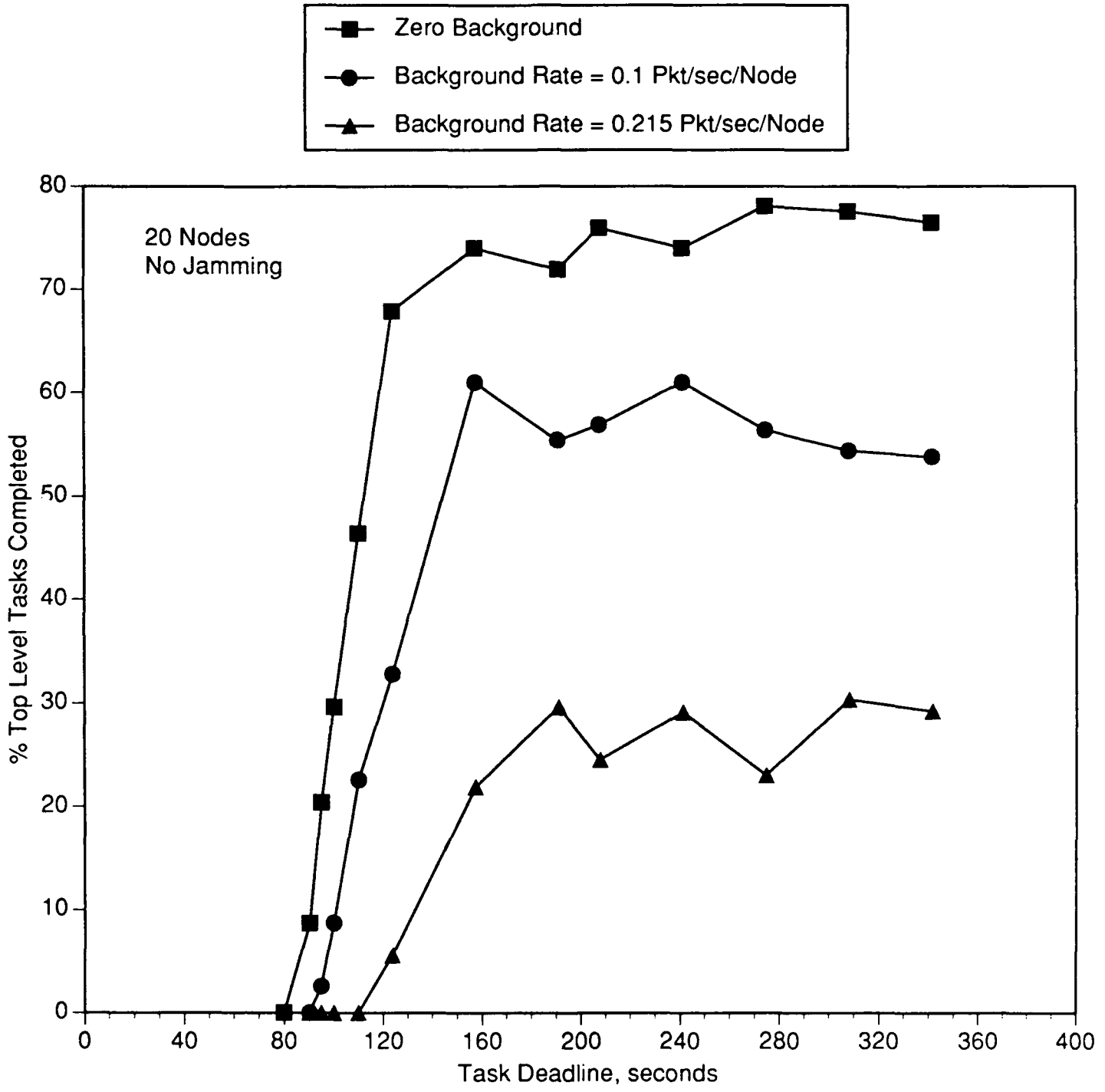


Figure. 12. Task completion rate Vs deadline time (without redundant processing)

fluctuations about the saturation level are due to statistical noise in the relatively small sample of top level task instantiations.

Figure 13 presents the results when redundant processing has been included. For these runs, the subtasks at the third level of the task hierarchy have been implemented redundantly. The shape of these curves is generally the same as before but, as expected, the saturation levels are raised.

5.3 Jamming

The above discussed results did not include any jamming effects. The jammer model as implemented has a number of different parameters that may be varied (see Appendix I): number of frequency bins (q), number of bins jammed when the jammer is on (r), the probability the jammer is on in any particular bin (pj), and the the Reed-Solomon error correction parameters (n,k). For the purpose of investigating jamming pj was selected as the variable, and the remaining parameters were fixed at $q = 1000$ frequency bins, $r = 500$ bins jammed, $n = 64$, and $k = 40$ (out of 64 bits, 40 were information bits). The jamming environment was uniform over the entire network.

While the model includes a suitable packet overhead including error correction, the number of packets shown in the task diagrams is the number of packets produced at the network level. The network packet length was 1024 bits; half was overhead.

Figure 14 shows the effects of jamming on top level task completion. Curves are presented for redundant and non-redundant operation. A background rate of 0.1 packets per second per node was included, and the deadline was 400 seconds (that is, well into the saturation region of the previous results). The jamming probability pj was varied from zero to 0.12. Below a value of about 0.08 jamming has little effect. Beyond this value, performance drops rapidly and beyond a value of 0.12, jamming is totally effective. For these particular set of conditions, redundant operation appears to be beneficial, but this aspect will be discussed further in what follows.

It should be noted that in the jamming model the parameters q, r and pj always appear as the term $pj\ (r/q)$. Thus, the results presented here are valid for other combinations of these three values.

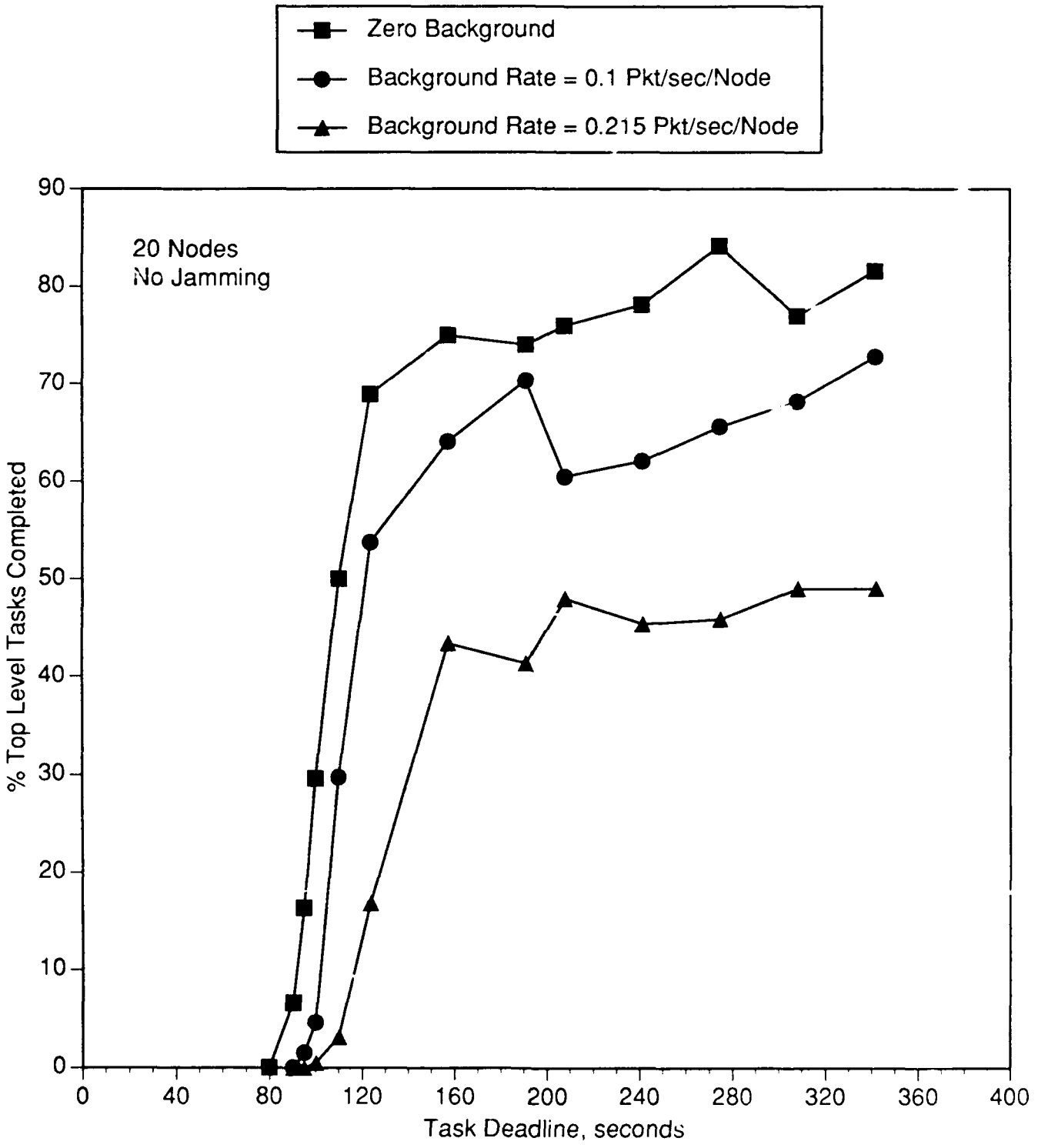


Figure. 13. Task completion rate Vs deadline time (with redundant processing)

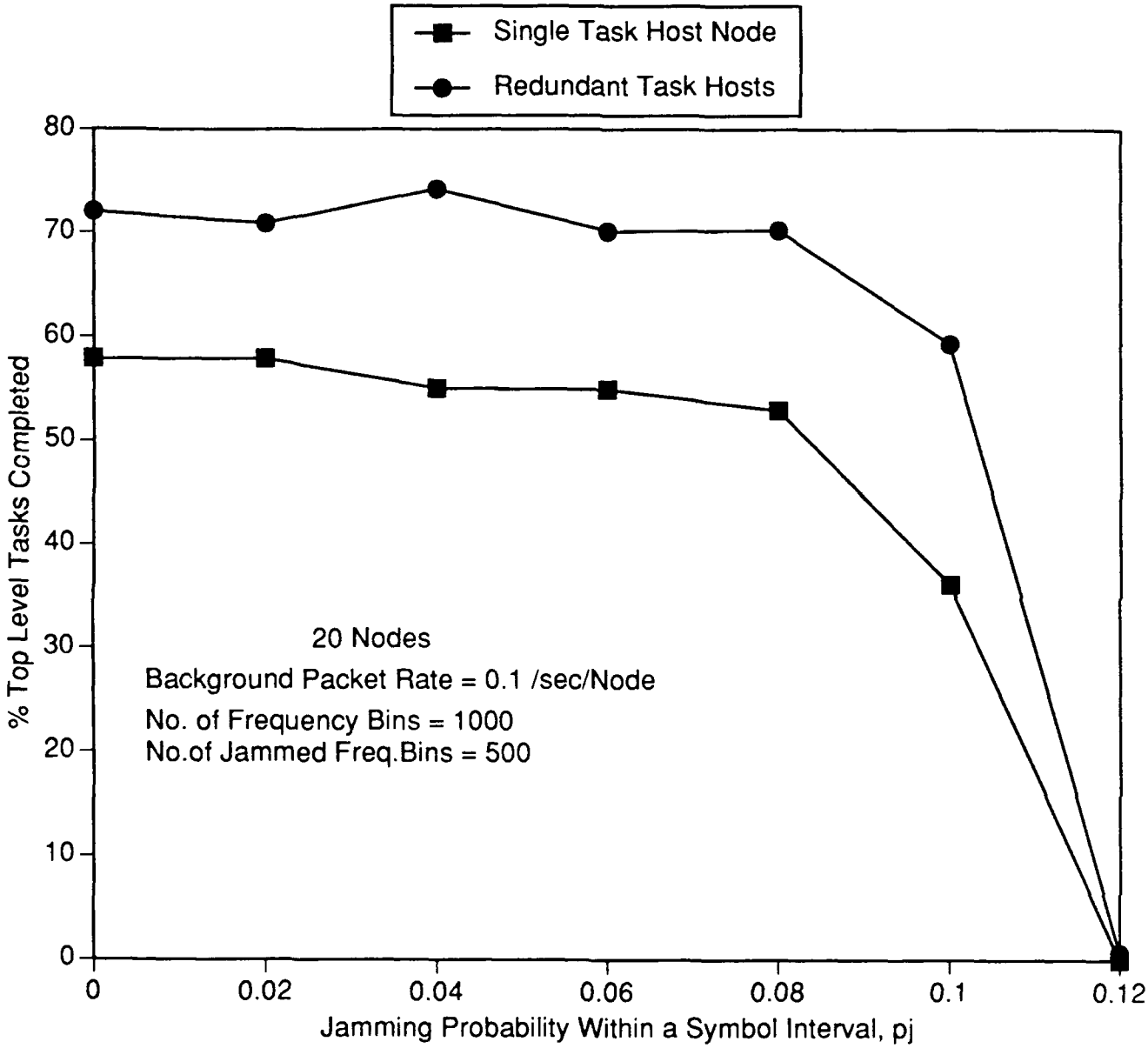


Figure 14. Task completion rate as a function of jamming probability, with and without redundant processing

5.4 Hierarchical Layers

Task distribution offers the possibility of increased throughput through parallel processing. By sharing the processing load with two or more nodes the time to complete a task may be potentially reduced. The advantages of parallelism may be offset to varying extent by the requirement to communicate the partial results among the cooperating nodes.

We consider a task that requires 200 seconds of processing with a 130 second deadline. While it is impossible to accomplish this task in a single node, by decomposing the task into two subtasks, each with 100 seconds of processing, it may be possible to satisfy the deadline requirement. Additional decomposition may provide even better performance.

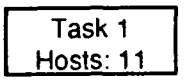
The task structure is illustrated in Figure 15 which shows the task decomposed into one, two, three, and four layers. The task has been deployed on the 20 node network described previously, and the nodes which host the various subtasks are indicated. In addition, the required inter-task communication is indicated by number of packets.

Note that in the two layer version, the processing time is divided between the two subtasks (100 seconds each) but the computational results - 40 packets - must be communicated by each subtask to the top level host. Similarly, in the three layer version, the processing time divided among four nodes at the third layer has been further subdivided to 50 seconds each, but 20 packets must be sent by each of the four to the two nodes in the second layer. These, in turn, must each send 40 packets to the top level host. Finally, in the four layer version, the processing load is reduced to 25 seconds per node, but each of the eight nodes in this layer must send 10 packets to the layer above, and so forth. In summary, while the addition of each layer halves the processing time, the communication between layers is increased which, in addition to processing, must be accomplished within the allotted deadline.

Simulation runs have been made on the two, three and four layer versions of this task (the one layer version fails trivially), with each run generating 100 instantiations of the top level task. The results are shown in Figure 16 which presents three bar charts showing (1) percent of tasks completed, (2) average task completion time, and (3) network traffic in terms of total number of packets generated.

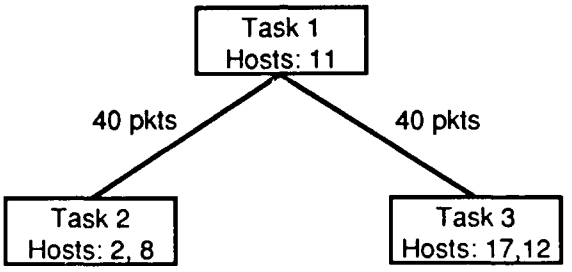
As may be expected, it can be seen from the number of packets generated, the increase in number of subtask layers is accompanied by a significant

One Layer



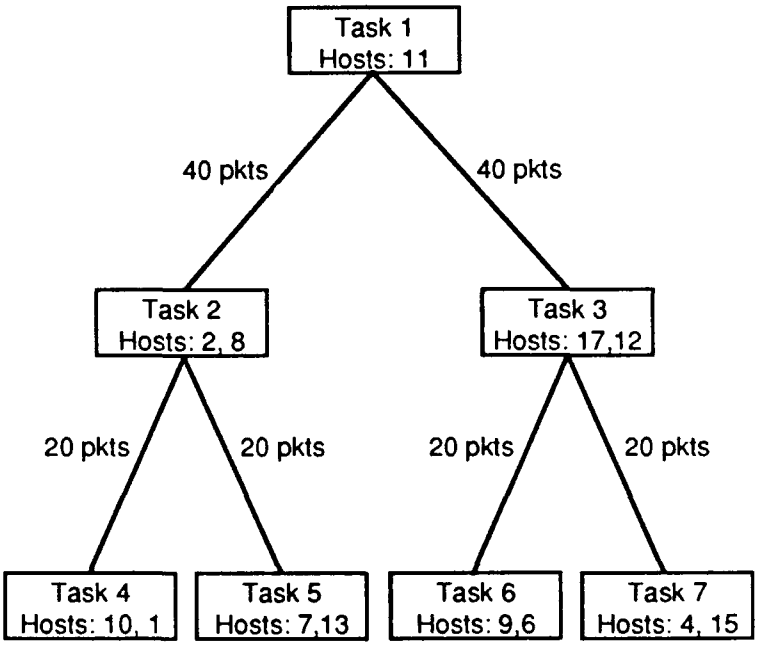
Processing Time = 200 seconds
Deadline = 130 seconds. (!)

Two Layers



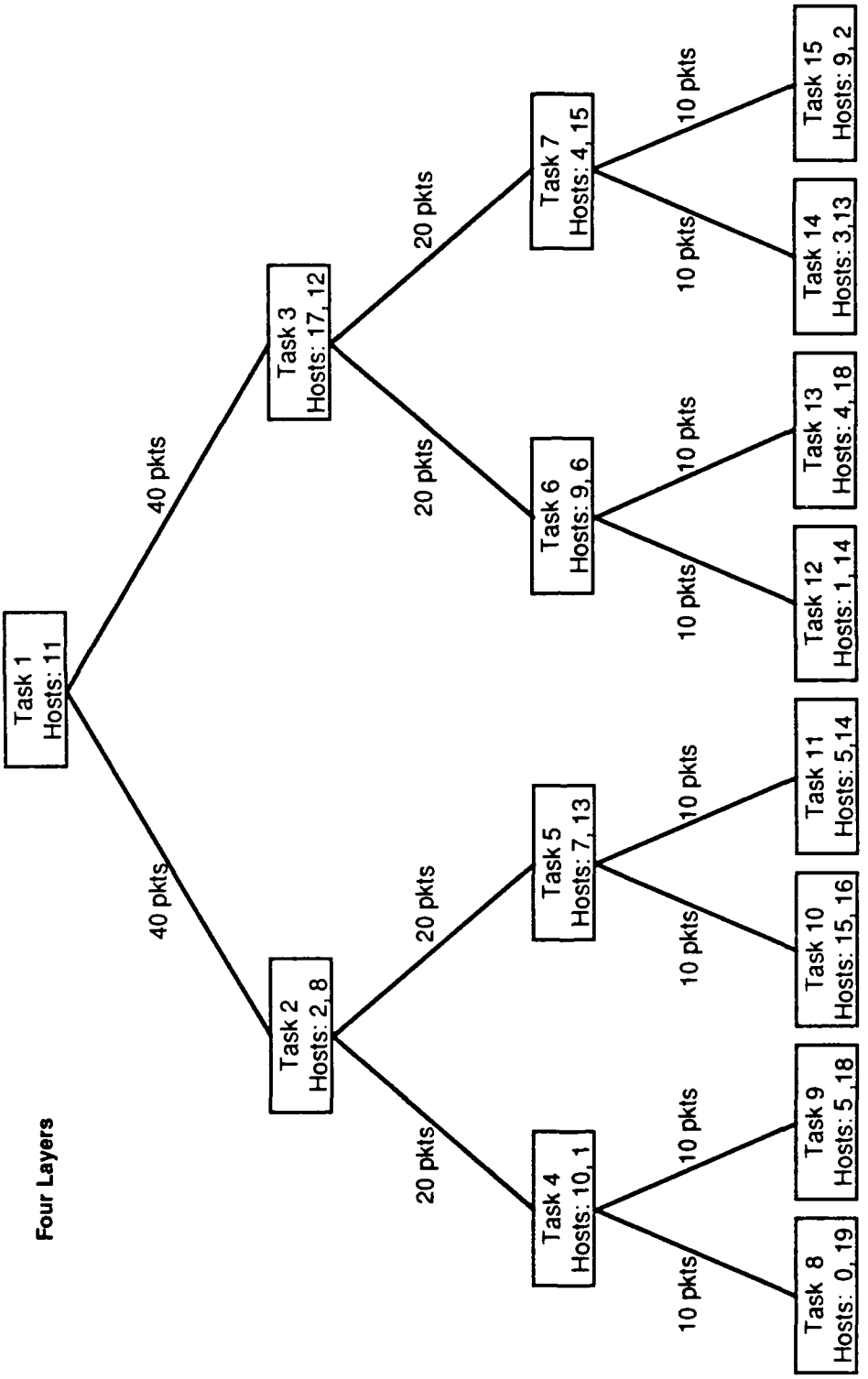
Processing Time = 100 seconds each node in layer 2.
Deadline = 130 seconds.

Three Layers



Processing Time = 50 seconds each node in layer 3.
Deadline = 130 seconds.

Figure 15 (a). Task decomposition into layered subtasks; one, two, and three layers. The model is used to obtain the results shown in Figure. 16.



Processing Time = 25 seconds each node in layer 4.
Deadline = 130 seconds.

Figure 15 (b). Task decomposition into layered subtasks; four layers.
The model is used to obtain the results shown in Figure. 16.

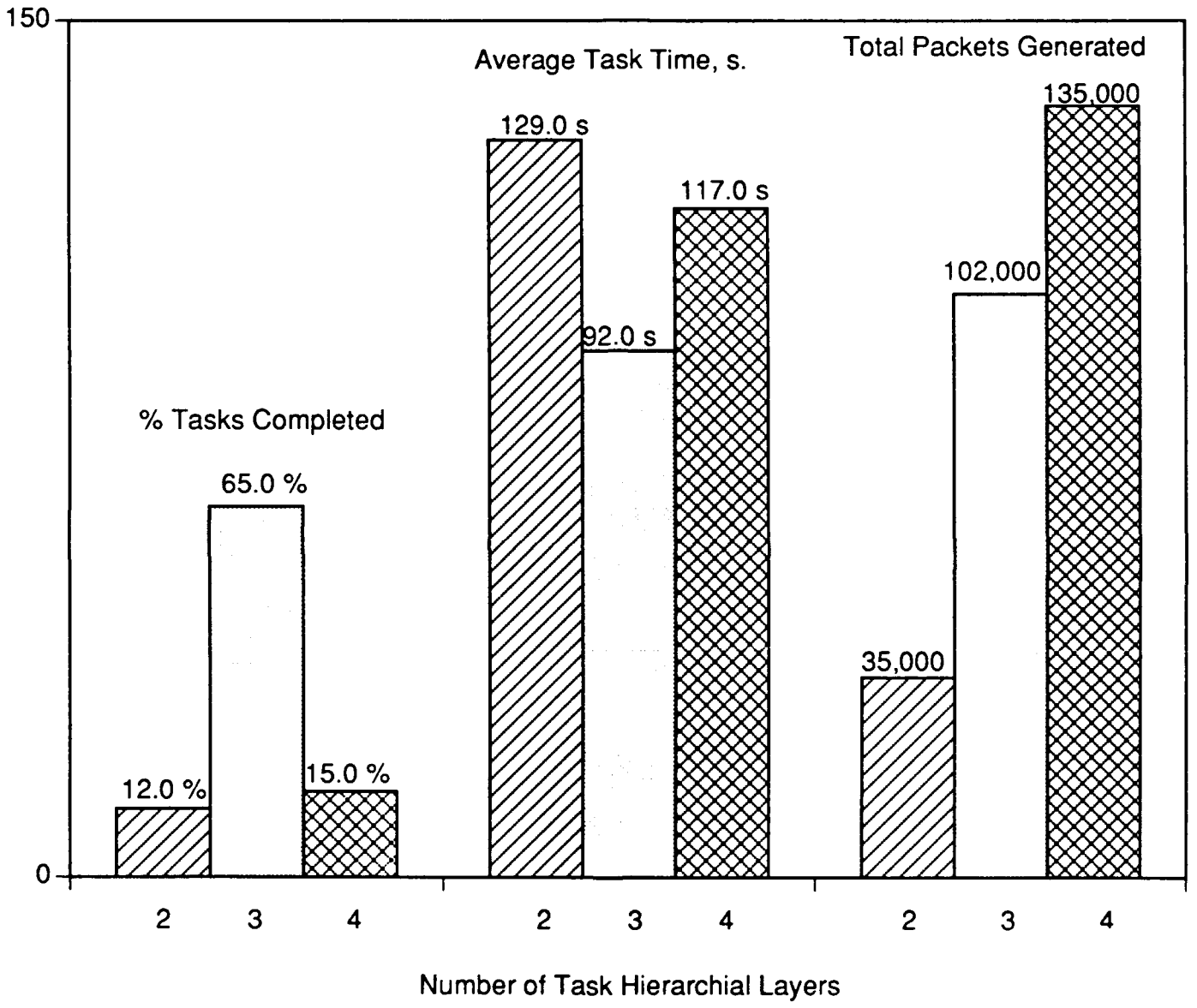


Figure 16. Performance as a function of number of layers (2, 3, and 4) of task decomposition. Refer to Figs 15 (a) and 15 (b) for the corresponding task structures used.

increase in network traffic. From the percent of tasks completed, it can be observed that going from one layer (not shown, but zero tasks completed) to two layers provides a modest improvement of 12% tasks completed. A more impressive improvement is afforded by the three layer version, i.e., 65% completed. However, the performance of the four layer version is worse, 15% completed. It is clear that the communication load is finally more than offsetting the improvement afforded by parallelization in the four-layer version. The task time to complete is consistent with these results, showing an optimum at three layers.

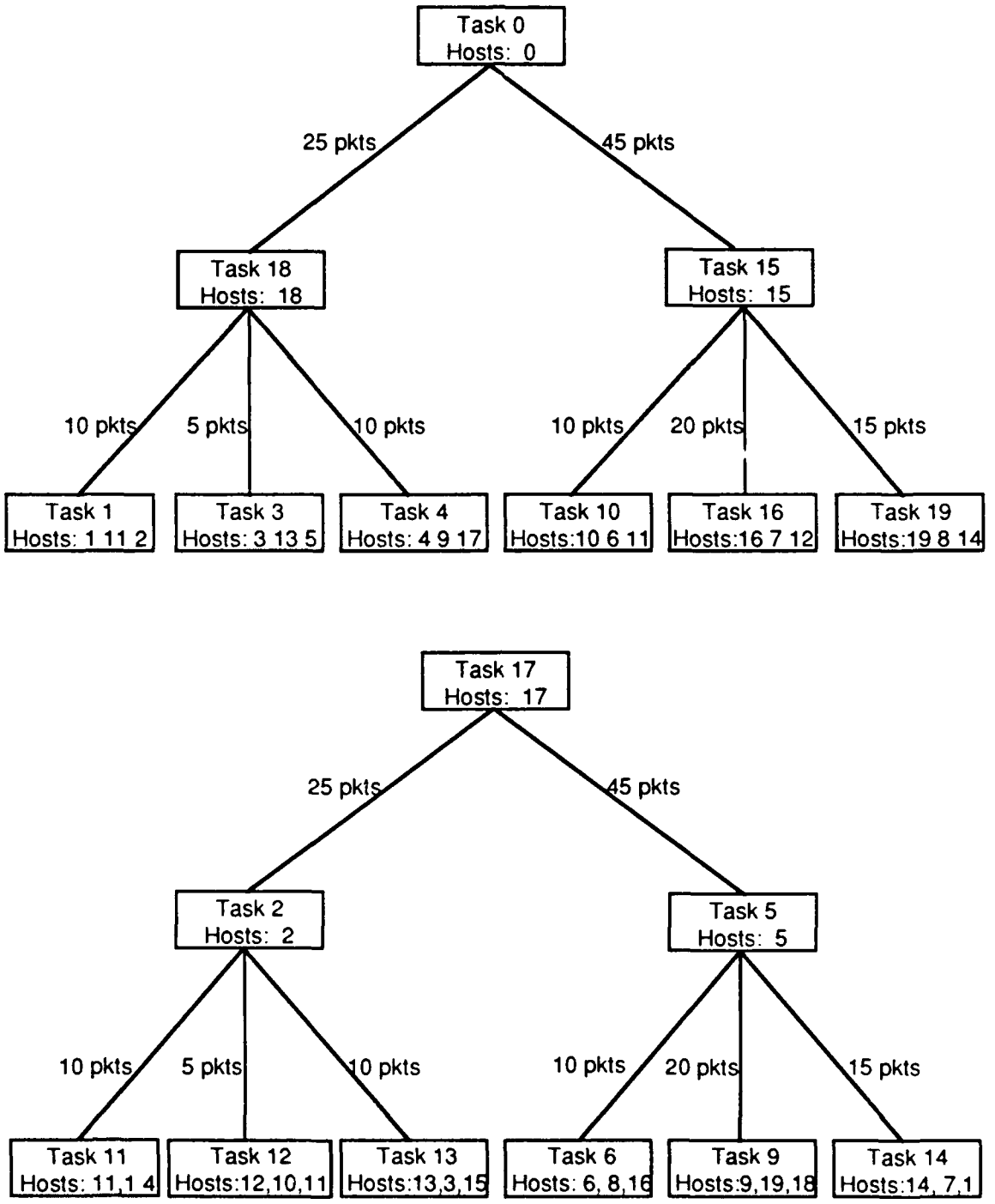
5.5 Task Redundancy

Because of the hierarchical structure, failure of any subtask, due to communication failure or deadline, will cause the entire task to fail. Redundant operation, that is, initiating some or all subtasks on multiple hosts, can improve task reliability. The added communication congestion due to these additional tasks can offset the advantages of redundancy.

To investigate this effect, we have varied redundancy level - zero (one host/task), single (two hosts/task), and double (three hosts/task) - in two quite different scenarios. In one scenario, single redundancy improved performance while double redundancy resulted in some deterioration. In the second scenario, additional redundancy always produced a negative effect.

The first case is based upon the task structure shown in Figure 17. This structure includes two similar three-layer hierarchies deployed on the 20 node network described previously. The nodes which host the various subtasks are indicated in the figure. Again, the required inter-task communication is indicated by number of packets. Note that, in the third layer only, three host nodes are specified for each subtask. The nodes were utilized in the order indicated: for no redundancy the first listed node was used, for single redundancy the first two nodes were used, and for double redundancy all three nodes were used. A deadline time of 200 seconds was imposed on these tasks.

Simulation runs were made for all the levels of redundancy. In all cases the two top level tasks are initiated simultaneously and in each run the pair were instantiated 100 times providing a sample size of 200. The results for the three levels of redundancy are presented in Figure 18 which provides three bar charts showing percent of tasks completed, total number of packets delivered, and total packets generated, shown correspondingly by the notation 1,2, and 3 in the figure.



Processing Time = Uniform (5, 10) Seconds for each task.

Figure 17. Task structure I used for redundant processing results shown in Fig. 18

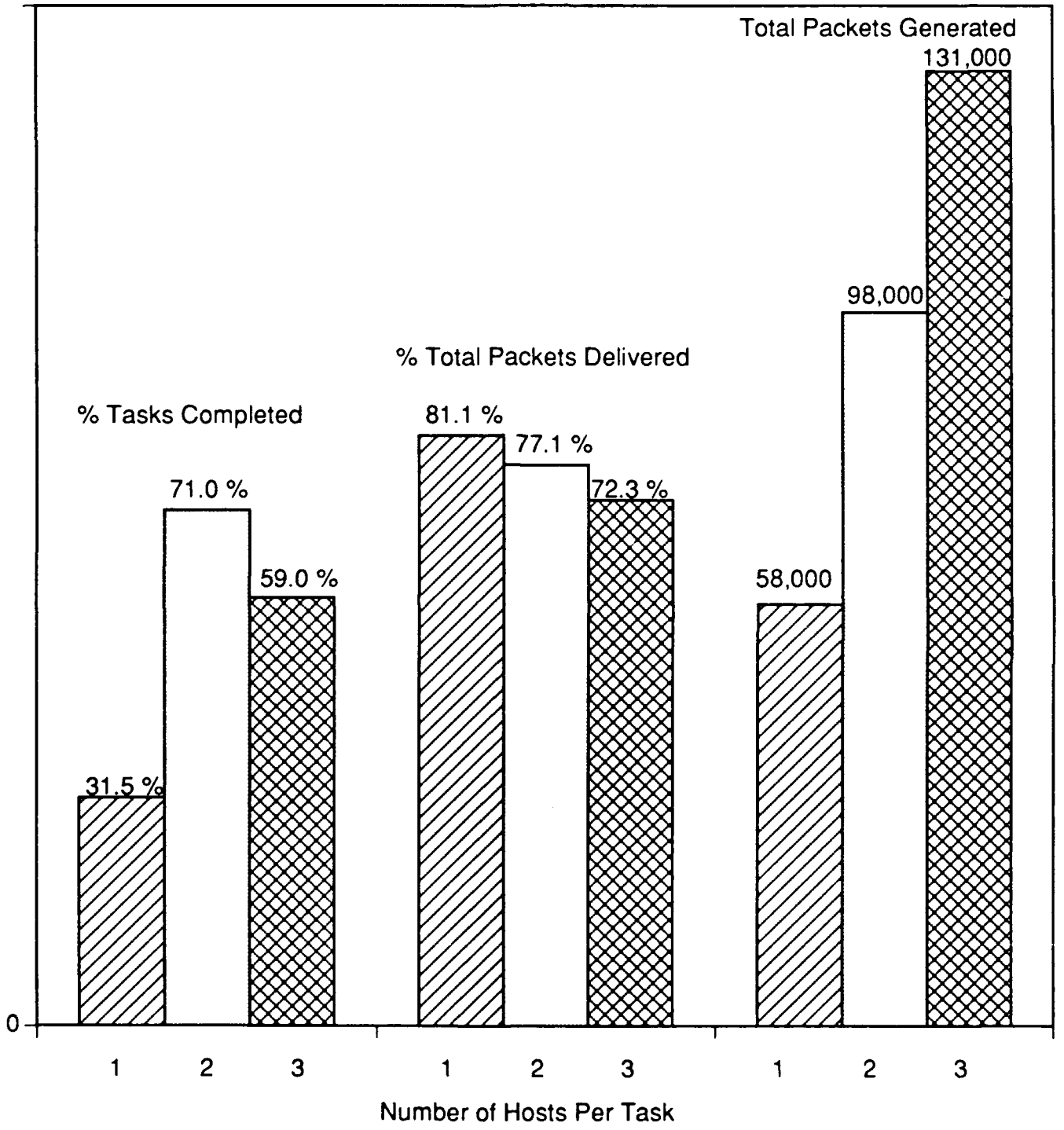
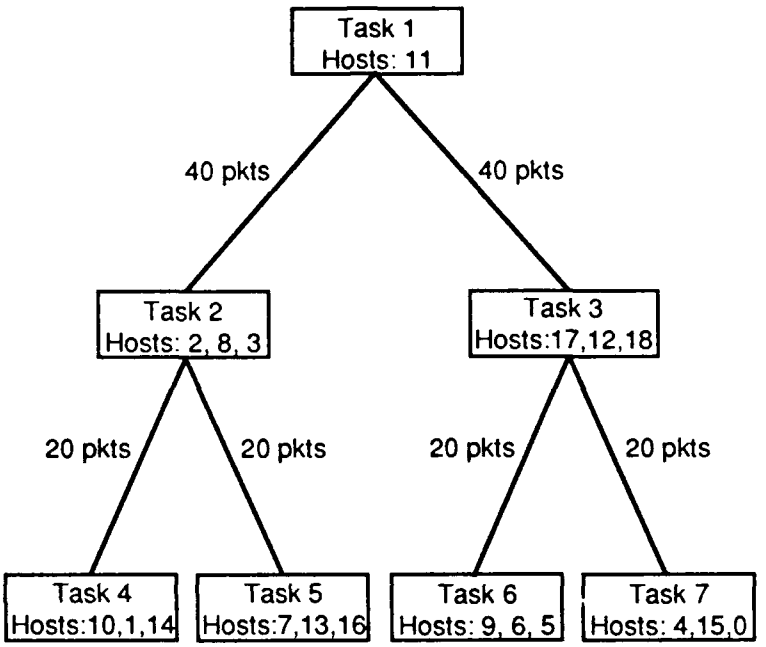


Figure 18. Effect of Redundant Processing on the Performance Parameters. Corresponding task structures are shown in Figure 17.

These results show the effect of redundancy on the underlying network; as expected increased redundancy results in more packets generated (58 to 98 to 131 thousand) and a lower percentage of packets delivered (81 to 77 to 73 percent). The deterioration in network performance results in when increased communication delays and failures offset the advantages of redundancy. Thus, while the percent tasks completed significantly improved in going from zero redundancy to single redundancy (31.5% to 71.0%), double redundancy not only fails to provide additional improvement, but in fact is somewhat worse (59.0%), though still significantly better than the case for no redundancy.

The second situation was run on the task structure shown in Figure 19, which indicates three host nodes for each subtask. As before, the nodes were used in the order indicated. The results are shown in Figure 20. Again, these results show the relationship between network performance and increased level of redundancy. In this case, redundancy is insufficient to offset loss of network performance. As redundancy is increased from zero, to single, and to double, completed tasks are reduced from 99, to 75, and to 38 percent respectively.



Processing Time = 50 seconds each node in layer 3.
Deadline = 130 seconds.

Figure 19. Task structure II used for redundant processing results shown in Fig. 20.

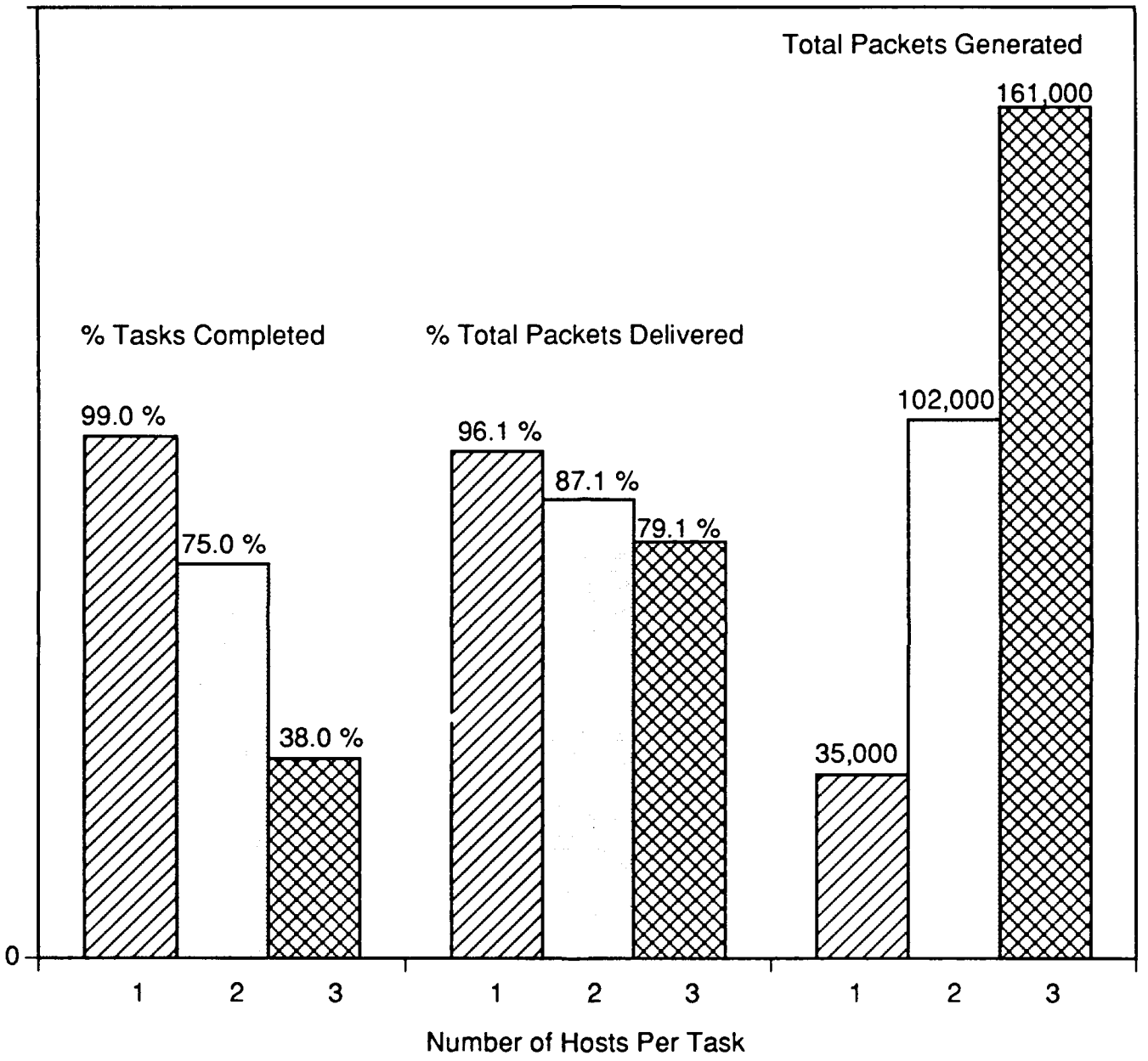


Figure 20. Effect of Redundant Processing on the Performance Parameters. Corresponding task structure is shown in Figure 19.

6. CONCLUDING REMARKS

The above results indicate that the model can be used as a design tool in evaluating the effectiveness of the communications support for carrying out the desired level of task processing, with the selection of the appropriate underlying parameters. Since usually the focus of the evaluation would be from an application point of view, analyzing the reasons for a given level of performance or its degradation in an integrated way from the link level and above is sometimes quite difficult. By introducing the constructs that allow tasks and subtasks to abort due to a time deadline, we have enhanced the operational realism of the model. With the facility to assign a subtask to two or more nodes concurrently for redundant processing, mission survivability may be improved through reduced task failures in the face of communication problems or node attrition.

In the work described in this report, we basically described a methodology to represent the interrelationships between processing and communications, developed a comprehensive simulation model to incorporate the methodologies, and derived quantitative results to demonstrate the relationships. The input data on network parameters and distributed task structure were chosen mainly to demonstrate the usefulness of the model as a performance evaluation and design tool. It will be interesting to apply the model to a real scenario if such data were to be available. This might require some changes to the existing input/output data formats in the simulation model.

Our planned future program envisages enhancement of the simulation tool and development of appropriate theoretical models to investigate the following:

(a) Scheduling Time-Critical Tasks Using Value-Based Approach
(b) Task allocation in Redundant Processing
(c) Modeling Multi-media Packet Radio Networks
(d) Applications of Knowledge-Based Network Management

ACKNOWLEDGMENTS

The authors thank Dr. Joseph Baker and Dr. Loren Clare for their help in determining parameters relationships for computing error rates in the presence of multiple access noise and jamming that were incorporated in the simulation model and for a number of useful discussions. We also thank the Army Research Office for supporting this work.

PUBLICATIONS, PERSONNEL AND INVENTIONS

I. Publications

1. R. J. Doyle, I. Shahnawaz, and A. R. K. Sastry, "A Simulation Model for Evaluation of Distributed Processing in Multi-hop Packet Radio Networks," (to be presented at the IEEE MILCOM'91, November 1991).

2. R. J. Doyle, I. Shahnawaz, and A. R. K. Sastry, "Impact of Communication Delays, Deadline Times, and Redundant Processing on Distributed Processing in Multi-hop Packet Radio Networks," (in preparation).

II. Scientific Personnel

Principal Investigator: A.R.K. Sastry

Co-investigators: Robert Doyle
Iftikhar Shahnawaz
Joseph Baker
Loren Clare

Degrees Earned: Not Applicable

III. Inventions: None

REFERENCES

[1]. International Standards Organization, ``Information Processing Systems - Open Systems Interconnection - Basic Reference Model," International standard, ISO/IS 7498, 1983.

[2]. ISO/TC97/SC21/WG7 N4883-4888, "Information Processing Systems - Open Systems Interconnection (OSI) - Open Distributed Processing ODP), June 1990.

[3]. C. Chabernaud and B. Vilain, "Telecommunication Services and Distributed Applications," IEEE Network Magazine, pp. 10-13, November 1990.

[4]. R.E. Kahn, S.A. Gronemeyer, J. Burchfiel, and R.C. Kunzelman, ``Advances in Packet Radio Technology," Proc.IEEE, vol.66, pp.1468-1496, November 1978.

[5]. B.M. Leiner, D.L. Nielson, and F.A. Tobagi, ``Issues in Packet Radio Network Design," Proc. IEEE, pp.6-20, January 1987.

[6]. J. Jubin and J.D. Tornow, ``The DARPA Packet Radio Network Protocols," Proc. IEEE, pp.21-32, January 1987.

[7]. W.C. Fifer and F.J. Bruno, ``Low Cost Packet Radio," Proc. IEEE, Vol.75, pp.33-42, January 1987.

[8]. N. Shacham and J. Westcott, ``Future Directions in Packet Radio Architectures and Protocols," Proc.IEEE, pp.83-99, January 1987.

[9]. M.S. Frankel, C.J. Graff, L.U. Dworkin, T.J. Klein, and R.L.desJardins, ``An Overview of the Army/DARPA Distributed Communications and Processing Experiment," IEEE Journal on Selected Areas in Communications, Vol.SAC-4, No.2, pp.207-215, March 1986.

[10]. K.W. Fertig, A.P. Andrews, and C-Y Wang, ``Knowledge-Based Management and Control of Communication Networks," Conf. Record, IEEE MILCOM, Monterey, CA, October 1986.

[11]. C-Y. Wang, M.W. Atkinson, K.W. Fertig, and A.R.K. Sastry, ``Performance Evaluation of Multi-hop Packet Radio Networks Using Simulation," Conf. Record, IEEE MILCOM, Monterey, CA, October 1986.

[12]. F. A. Tobagi, ``Modeling and Performance Analysis of Multi-hop Packet Radio Networks,'' Proc.IEEE, pp.135-155, January 1987.

[13]. N. Abramson, ``The throughput of packet broadcasting channels,'' IEEE Trans on Commun., pp117-128, January 1977.

[14]. R.M. Metcalfe and D.R. Boggs, ``ETHERNET: Distributed packet switching for local computer networks,'' Communications of the ACM, pp. 395-403, 1976.

[15]. L. Kleinrock and S.S. Lam, ``Packet switching in a multiaccess broadcast channel: Performance evaluation,'' IEEE Trans. on Commun., pp. 410-423, April 1975.

[16]. S.S. Lam and L. Kleinrock, ``Packet Switching in a Multiaccess Broadcast Channel: Dynamic Control Procedures,'' IEEE Trans. on Commun., pp. 891-904, September 1975.

[17]. F. Tobagi and L. Kleinrock, ``Packet switching in radio channels: Part IV- Stability considerations and dynamic control in carrier sense multiple access,'' IEEE Trans. on Commun., pp. 1103-1120, October 1977.

[18]. G. Fayolle, E. Gelenbe, and J. Labetoulee, ``Stability and Optimal Control of the Packet switching Broadcast Channels,'' J. of the Asso. for Comput. Mach., pp. 375-386, July 1977.

[19]. B. Hajek and T. vanLoon, ``Decentralized dynamic control of a multiaccess broadcast channel,'' {IEEE Trans. Automat. Control, vol. AC-27, no. 3., pp.559-569, June 1982.

[20]. L.P. Clare, ``Delay analysis of stable slotted ALOHA systems,'' Proc. INFOCOM 86, pp. 10-19, Miami, Florida, April 1986.

[21]. L.P. Clare, ``Control Procedures for Slotted ALOHA Systems that Achieve Stability,'' Proc. ACM SIGCOMM '86 Symposium on Communication Architectures and Protocols, pp. 302-309, Stowe, Vermont, August 1986.

[22] L.P. Clare, ``On the Performance of a Class of Slotted ALOHA Dynamic Control Procedures,'' Proc. 25th IEEE Conf. on Decision and Control, Athens, Greece, December 1986.

[23]. L.P. Clare, J.E. Baker, and A.R.K. Sastry, "A Performance Comparison of Control Policies for Slotted ALOHA Frequency-Hopped Systems," Conf. Record, pp. 608-614, IEEE MILCOM'90, Monterey, October 1-3, 1990.

[24]. L.P. Clare and J.E. Baker, "The Effects of Jamming on Control Policies for Frequency-Hopped Slotted ALOHA," Conf. Record, pp. 1132-1138, IEEE GLOBECOM, San Diego, December 2-5, 1990.

[25] L.P. Clare and J.E. Baker, "Robust Procedures for Implementation of Dynamic Control Policies for Frequency-hopped Slotted ALOHA," (To be presented at the IEEE MILCOM, November 1991)

[26]. R. L. Pickholtz, D. L. Schilling, and L. B. Milstein, ``Theory of Spread Spectrum Communications,'' IEEE Trans. on Communications (PartI, Sp. issue on spread-spectrum communications), pp 855-884, May 1982.

[27]. M.B. Pursley, ``The Role of Spread Spectrum in Packet Radio Networks,'' Proc.IEEE, pp.116-134, January 1987.

[28]. A. Ephremides, J.E. Wieselthier, and D.J. Baker, ``A Design Concept for Reliable Mobile Radio Networks with Frequency Hopping Signaling,'' Proc. IEEE, pp.56-73, January 1987.

[29]. C. L. Weber, G. K. Huth and B. H. Batson, ``Performance Considerations of Code Division Multiple Access Systems,'' IEEE Tran. on Veh. Tech. vol. VT-30, pp 3-9, February 1981.

[30]. D. H. Davis and S. A. Gronemeyer, ``Performance of Slotted ALOHA Random Access with Delay Capture and Randomized Time of Arrival,'' IEEE Trans. on Commun., vol. COM-28, pp 703-710, May 1980.

[31]. D. Raychaudhuri, ``Performance Analysis of Random Access Packet Switched Code Division Multiple Access Systems,'' IEEE Trans. on Commun., vol. COM-29, pp 895-901, June 1981.

[32]. M. Kavehrad, ``An Accessing Technique for Information Packet Networks,'' EASCON Record, pp. 42-45, 1981.

[33]. J. M. Musser and J. N. Daigle, ``Throughput Analysis of an Asynchronous Code Division Multiple Access (CDMA) System,'' in Conf. Rec., IEEE Int. Conf. on Commun., pp 2F-2-1-2F-2-7, June 1982.

[34]. P. Economopoulos and M.L. Molle, ``On the Performance of Slotted ALOHA in a Spread Spectrum Environment,'' Symposium Record, ACM SIGCOMM Symposium on Communication Architectures and Protocols, pp.234-241, June 1984.

[35]. A. R. K. Sastry, ``Effect of Acknowledgement Traffic on the Performance of Slotted ALOHA-Code Division Multiple Access Systems,'' in Conf. Rec., IEEE MILCOM, pp 68-74, Sept. 1983 (Also, a shorter version in IEEE Trans. on Commun., pp~1219-1222, November 1984).

[36]. R. A, Scholtz, ``Optimal CDMA codes,'' Conf. Rec., IEEE Nat. Telecomm. Conf., pp. 54.2.1-54.2.4, 1979.

[37]. R. Gold, ``Optimal Binary Sequences for Spread Spectrum Multiplexing,'' IEEE Trans. on Inform. Theory, pp. 619-621, October 1967.

[38] W. Kim and W.Stark, "Optimum rate {Reed}-{Solomon} Codes for Frequency-hopped Spread-Spectrum Multiple-Access Communication Systems}, IEEE Trans. on Communications, pp. 138-144, 1989.

[39]. S. Davidovici, L. B. Milstein, and D. L. Schilling, ``A New Rapid Acquisition Technique for Direct Sequence Spread Spectrum Communications,'' IEEE Trans. on Commun., pp. 1161-1168, November 1984.

[40]. M. S. Frankel, ``Advanced Technology Testbeds for Distributed, Survivable Command, Control, and Communications (C^3),'' Conf. Rec., IEEE MILCOM, pp10.2-1-10.2-13, 1982.

[41]. B.H. Davies and T.R. Davies, ``The Application of Packet Switching Techniques to Combat Net Radio,'' Proc. IEEE, pp.43-55, January 1987.

[42]. L.P. Clare and A.R.K. Sastry, "The Effects of Slotting, Burstiness, and Jamming in Frequency-Hopped Random Access Systems," Conf. Record, pp. 154-160, IEEE MILCOM'89, Boston, October 15-18, 1989.

[43]. F. A. Tobagi, ``Analysis of a Two-Hop Centralized Packet Radio Network-Part I: Slotted ALOHA,'' IEEE Trans. on Commun., pp.196 -207, February 1980.

[44]. H. E. de Pedro, and J. K. Wolf, ``On the Design of Spread Spectrum Communication Networks,'' Conf. Rec., IEEE Int. Conf. on Commun. (ICC), pp. 15.1.1-15.1.3, June 1980.

[45]. B. M. Leiner, ``A Simple Model for Computation of Packet Radio Network Communication Performance,'' IEEE Trans. on Commun., pp. 2020 -2023, December 1980.

[46]. T.C. Hou and V. O. K. Li, ``Performance Analysis of Routing Strategies in Multi-hop Packet Radio Networks,'' Conf. Rec., IEEE GLOBECOM, pp. 487-492, November 1984.

[47]. L. Kleinrock and J.Silvester, ``Spatial Reuse in Multihop Packet Radio Networks,'' Proc.IEEE, pp.156-167, January 1987.

[48]. J.R. Agre, I. Shahnawaz, and A.R.K. Sastry, "Transport Protocol Parameters in a Multiple LAN Environment," Conf. Record, IEEE Int. Conf. on Communications, pp. 1189-1194, June 1988.

APPENDIX I
Error Model for the Packet Radio Network with Jamming

1. System Model

The the system consists of a collection of N packet radio stations (transceivers). For station j, we define the hearing set H_j to be the set of stations whose transmissions are heard at station j. The stations use slow frequency hop (one symbol per hop) spread spectrum communications. The signal hops among q frequency bins. Stations are not synchronized. Packets are encoded using a Reed-Solomon code. We will calculate error probabilities below for both erasure channels and channels with no side information.

For the CDMA-ALOHA operation, receiver directed codes are used. If a message is transmitted to a station, and another message is already being transmitted to the station, or another message starts transmission within a time window t_c, the first message will fail.

2. The Jamming Model

We consider an on-off partial band tone jammer. That is, the jammer switches between on and off states. While on, the jammer completely blocks a group of frequency bins at a subset of the total collection of receivers. We assume that the jammer switches from the on state to the off state and back according to a memoryless slotted process, with slot size equal to the dwell interval. However, the jammer slotting is not synchronized with the slotting at the receiver. The frequency bins which are jammed are assumed to be an arbitrary and uniformly distributed subset of the bins.

The jammer is described by:

(i) the collection of receivers jammed (possibly varying),
(ii) r: the number of frequency bins jammed when on,
(iii) p_j, the probability that the jammer is on in any particular slot (we only require that this probability stay constant during the transmission of a tagged packet), and
(iv) any power limitations.

3. Probability of Correct Reception

We focus upon a particular message transmission. We assume that the the following information is known:

(i) R_S : the number of ongoing transmissions directed to the receiver of the tagged message (excluding the tagged message), measured at the start of transmission;
(ii) R_C : the number of ongoing transmissions directed to the receiver of the tagged message (excluding the tagged message), measured at the end of the capture window;
(iii) N_S : the total number of transmissions in progress initiated by members of the receiver's hearing set (regardless of their destination, and excluding the tagged message), measured at the start of transmission of the tagged message;
(iv) N_e : the total number of transmissions in progress initiated by members of the receiver's hearing set (regardless of their destination and excluding the tagged message), measured at the end of transmission of the tagged message.

If either $R_S > 0$ or $R_C > 0$, the tagged message fails with probability one.

Otherwise, let $\tilde{N} = \max \{N_S , N_e\}$. We will approximate the (varying) number of interfering transmissions by assuming that $\tilde{N}$ transmissions interfere throughout the tagged transmission. The tagged message is n symbols long, and has been encoded using an (n,k) Reed-Solomon code.

During the course of the transmission, the jammer interferes with the receiver with probability p_j in each (jammer) slot.

Consider an arbitrary symbol (dwell interval) of the tagged message. Since the stations are not synchronized, the probability that an interfering transmission misses the bin occupied by the tagged message is

$$\left(\frac{q-1}{q}\right)^2 .$$

The probability that the jammer misses is

$$\left(1 - p_j \frac{r}{q}\right)^2 .$$

Thus, the probability that an arbitrary symbol of the tagged message is hit, given $\tilde{N}$ interfering transmissions, is

$$P_{h,\tilde{N}} = 1 - \left(\frac{q-1}{q}\right)^{2\tilde{N}} \left(1 - p_j \frac{r}{q}\right)^2$$

We now make the standard approximating assumption of independence between the dwell intervals. If perfect side information is available, the probability of codeword (packet) error is given by [38]

$$P_e(\tilde{N},n,k,q) = \sum_{i=n-k+1}^{n} \binom{n}{i} P_{h,\tilde{N}}{}^{i} \left(1-P_{h,\tilde{N}}\right)^{n-i} .$$

Now suppose that no side information is available. In this case, the probability of codeword error is bounded by

$$P_e(\tilde{N},n,k,q) = 1 - \sum_{i=0}^{(n-k)/2} \binom{n}{i} P_{h,\tilde{N}}{}^{i} \left(1-P_{h,\tilde{N}}\right)^{n-i} .$$

```
================================================================================
                              APPENDIX  II
================================================================================
                TYPICAL  OUTPUT  FROM  A  SIMULATION  RUN
================================================================================
Elapsed CPU Time = 912 Sec
Events Processed: 6391851

 Simulated Time                =      50143.67 sec.

------------------------- Link Layer (LL) Statistics --------------------------

 LL NOTES: (1) Generated Packets => Originals (including AL Msgs + TL Acks), and
                                    Background Pkts.
           (2) Delivered Pkts.   => The Originals that successfully reached their
                                    Destination nodes
           (3) Forwarded Pkts.   => Those Originals requiring INTERMEDIARY NODE
                                    relays;may or may not be their final Dstns.
           (4) Duplicate Pkts.Received => Duplicates of ALREADY RECEIVED Pkts.
           (5) Failed Pkts.      => Those that EXHAUSTED their LL RETRIES
           (6) Transmitted Pkts. => Orgnls.+Duplcts.+Bckgrnd.+LLRetries;which will
                                    end up either Recvd.,Clashed,Jammed or Ignored
           (7) Received Pkts.    => Include Pkts.NOT at final destn.+Frwrded.+Fail
           (8) Clashed Pkts.     => Transmitted Pkts. which were started WITHIN
                                    the COLLISION WINDOW of their Xceivers
           (9) Ignored Pkts.     => Transmitted Pkts. that 'found' their Destn.
                                    Xcvrs. PRE-OCCUPIED WITH ANOTHER RECEPTION
--------------------------------------------------------------------------------

 Total Packets Generated     =       101803           (See LL NOTE 1)
 Total Packets Delivered     =        92407 ( 90.8 %)(See LL NOTE 2)
 Total Packets Failed        =         9396 (  9.2 %)
 Total Packets Forwarded     =       109261           (See LL NOTE 3)
 Duplicate Packets Received  =       103642           (See LL NOTE 4)
 Local.Gen+Relayed Info.Pkts=        211064
 Packets' Retries Exhausted  =        27264 ( 12.9 %)(See LL NOTE 5)
 Total Acks. Generated       =       305310
 Total Acks. Received        =       194164 ( 63.6 %)
 Total Packets Acked.        =       183800
 Control Packets Failed      =            0
 Total Packets Transmitted   =       835381           (See LL NOTE 6)
 Total Packets Received      =       499474 ( 59.8 %)(See LL NOTE 7)
 Total Packets Clashed       =        17312 (  2.1 %)(See LL NOTE 8)
 Total Packets Jammed        =        74415 (  8.9 %)
 Total Packets Ignored       =       244180 ( 29.2 %)(See LL NOTE 9)
 Offered Load                =          6.5 %
 Utilization                 =          5.9 %
 Avg. Packet Total Delay     =      261.585 ms.
 Avg. Packet Delay Per Hop   =      133.330 ms.
 Avg. No. of Hops Per Pkt    =        2.095
```

---------- Transport Layer (TL) Statistics ----------

TL NOTE: T.P.D.U's = Transport Protocol Layer Data Units

Number of T.P.D.U's per Appln. Layer Message	=	20
Total Number of T.P.D.U's (+TLRetries) Generated	=	45295
Number of Failed (TLRetries Exhausted) T.P.D.U's	=	41 (0.1 %)
Avg. T.P.D.U. End to End Delay	=	921.337 ms.
Avg. T.P.D.U. End to End Delay Per Hop	=	461.697 ms.
Max. T.P.D.U. End to End Delay	=	11855.440 ms.

---------- Application Layer (AL) Statistics ----------

AL NOTE: With each SUB TASK there is an associated a REQUEST and RESPONSE ALMsg
NO such ALMsg association exists for a TOP LEVEL TASK.

Total SubTasks Created	=	1000
Total SubTasks Failed	=	41 (4.1 %)
Total SubTasks Timed Out	=	103 (10.3 %)
Total SubTasks Cmpltd.Late	=	821 (82.1 %)
Redundant SubTasks(Ignored)	=	213
Top Level Tasks Initiated	=	100
Top Level Tasks Completed	=	65 (65.0 %)
Top Level Tasks Timed Out	=	35 (35.0 %)
Top Level Tasks Aborted	=	0 (0.0 %)
Avg. Task Queueing Delay	=	0.0 ms.
Avg. Task Processing Delay	=	91833.1 ms.
Max. Task Processing Delay	=	123934.8 ms.

Number of TL-Transmissions

	Bin	Contents	Cumulative %
<=	0.000	0	0.0
	1.000	38521	85.1
	2.000	5012	96.2
	3.000	1179	98.8
	4.000	383	99.6
	5.000	121	99.9
	6.000	38	100.0

Samples	=	45254
Average	=	1.203
Maximum	=	6.000
Minimum	=	1.000
Std. Dev.	=	0.560

==========

```
Parameter File: paramsb0r_032Mecd_3b.dat
Node File:      node20.dat

Packet Radio Simulation Parameters   Wed Jul 31 16:32:25 1991

MaxTransRange          = 30.000000
PropagationSpeed       = 300000000.000000
CollisionWindowLength  = 156.250000
TransmissionRate       = 0.032000
InitPacingDelay        = 70000.000000
MaxCarrierSenseDelay   = 640.000000
BackgroundRate         = 0.000000
RTSwitchDelay          = 500.000000
FailedLinkProbability  = 0.100000
CntlPktInterval        = 0.000000
MgrInterval            = 0.000000
StatusReportInterval   = 0.000000
DisplayQueryInterval   = 0.000000
StartTrace             = 0.000000
TimeStampInterval      = 0.000000
Estimated Comm.Delay   = 0.300000
InfoPktLength          = 1024
CntlPktLength          = 256
AckPktLength           = 64
LinkPreamble           = 24
LinkPktOverhead        = 64
MaxRetries             = 6
RandomStreamSim        = 0
RandomStreamInp        = 1
SingleHostFlag         = FALSE
DeadlineFlag           = TRUE
ActiveAckFlag          = TRUE
AckDuplicatesFlag      = TRUE
MgrCommFlag            = FALSE
DebugFlag1             = FALSE
DebugFlag2             = FALSE
DebugFlag3             = FALSE


RecptFile:      recptq1Kr_5Kpj_1.dat

Jammer Parameters
No of Frequency Bins     = 1000
No of Bins Jammed        = 500
Probability Jammer On    = 0.100000
Total Bits (n)           = 64
Info Bits (k)            = 40
Side Info Available?     = FALSE
Output Trace On?         = FALSE
```

```
Transport Layer Input File: transXZ.dat

Transport Layer Parameters
Trnsprt.Layer Info.Pkt.Len.=     512
---''---''--  Pkt.Overhead =      64
Max.No.Pkt.Retransmissions =       6
Pkt.Xmission.Delay (usec.) = 2000000.000000
Delay Histogram No.of Bins =       0
Delay ---''--- Uppr.Lt(sec)=    10.000000
Delay/Hop Histo.No.of Bins =       0
Delay/Hop -''- Uppr.Lt(sec)=     2.500000


Task Input File: task.dat

Task 1
   Hosted at Node: 11
   Request Msg Length Type      : Constant( 0.000000 )
   Response Msg Length Type     : Constant( 0.000000 )
   Proc.Dly Type, Durn.(sec.)   : Constant( 2.000000 )

   SubTask Sequence: 1
      Seq.Delay, sec.: Uniform( 0.000000, 1.200000; 0 )
      Max. SubTask Time Out (Computed,if not input), sec.: 92.400000

      SubTask(s): 2
      Estimated SubTask(s) Total Time, sec.: 92.400000

   SubTask Sequence: 2
      Seq.Delay, sec.: Uniform( 0.000000, 1.200000; 0 )
      Max. SubTask Time Out (Computed,if not input), sec.: 98.700000

      SubTask(s): 3
      Estimated SubTask(s) Total Time, sec.: 98.700000

Task 2
   Hosted at Node: 2  8
   Request Msg Length Type      : Constant( 1.000000 )
   Response Msg Length Type     : Constant( 40.000000 )
   Proc.Dly Type, Durn.(sec.)   : Constant( 2.000000 )

   SubTask Sequence: 1
      Seq.Delay, sec.: Uniform( 0.000000, 1.200000; 0 )
      Max. SubTask Time Out (Computed,if not input), sec.: 64.600000

      SubTask(s): 4
      Estimated SubTask(s) Total Time, sec.: 64.600000

   SubTask Sequence: 2
      Seq.Delay, sec.: Uniform( 0.000000, 1.200000; 0 )
      Max. SubTask Time Out (Computed,if not input), sec.: 58.300000

      SubTask(s): 5
      Estimated SubTask(s) Total Time, sec.: 58.300000
```

```
Task 3
   Hosted at Node: 17  12
   Request Msg Length Type      : Constant( 1.000000 )
   Response Msg Length Type     : Constant( 40.000000 )
   Proc.Dly Type, Durn.(sec.)   : Constant( 2.000000 )

   SubTask Sequence: 1
      Seq.Delay, sec.: Uniform( 0.000000, 1.200000; 0 )
      Max. SubTask Time Out (Computed,if not input), sec.: 64.600000

      SubTask(s): 6
      Estimated SubTask(s) Total Time, sec.: 64.600000

   SubTask Sequence: 2
      Seq.Delay, sec.: Uniform( 0.000000, 1.200000; 0 )
      Max. SubTask Time Out (Computed,if not input), sec.: 70.900000

      SubTask(s): 7
      Estimated SubTask(s) Total Time, sec.: 70.900000

Task 4
   Hosted at Node: 10  1
   Request Msg Length Type      : Constant( 1.000000 )
   Response Msg Length Type     : Constant( 20.000000 )
   Proc.Dly Type, Durn.(sec.)   : Constant( 52.000000 )

   SubTask Sequences: None

Task 5
   Hosted at Node: 7  13
   Request Msg Length Type      : Constant( 1.000000 )
   Response Msg Length Type     : Constant( 20.000000 )
   Proc.Dly Type, Durn.(sec.)   : Constant( 52.000000 )

   SubTask Sequences: None

Task 6
   Hosted at Node: 9  6
   Request Msg Length Type      : Constant( 1.000000 )
   Response Msg Length Type     : Constant( 20.000000 )
   Proc.Dly Type, Durn.(sec.)   : Constant( 52.000000 )

   SubTask Sequences: None

Task 7
   Hosted at Node: 4  15
   Request Msg Length Type      : Constant( 1.000000 )
   Response Msg Length Type     : Constant( 20.000000 )
   Proc.Dly Type, Durn.(sec.)   : Constant( 52.000000 )

   SubTask Sequences: None
```

Appendix II

Exogenous Events Input File : exogYJ.dat

Event Schedule Time,sec.	Application Task Description	Task Id No.	Host Node No.	Task Igt Descr. Distbn.Type,Params.	Task Deadline (seconds)
0.0	jammerAction			ON	
0.0	applGen	11	1	Constant 500.0	130.0